Nuno Lourenço
Ricardo Martins
Nuno Horta

Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects

Springer

Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects

Nuno Lourenço · Ricardo Martins
Nuno Horta

Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects

Springer

Nuno Lourenço
Instituto de Telecomunicações, Instituto Superior Técnico
Universidade de Lisboa
Lisbon
Portugal

Nuno Horta
Instituto de Telecomunicações, Instituto Superior Técnico
Universidade de Lisboa
Lisbon
Portugal

Ricardo Martins
Instituto de Telecomunicações, Instituto Superior Técnico
Universidade de Lisboa
Lisbon
Portugal

ISBN 978-3-319-42036-3 ISBN 978-3-319-42037-0 (eBook)
DOI 10.1007/978-3-319-42037-0

Library of Congress Control Number: 2016945775

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG Switzerland

To Alina and Íris

Nuno Lourenço

To my little girls, Nádia, Joana and Daniela

Ricardo Martins

To Carla, João and Tiago

Nuno Horta

Preface

Over the past few decades, very large scale integration technologies have been widely improved, allowing the proliferation of consumer electronics and enabling the steady growth of the integrated circuit (IC) market to an estimated value of over $350 billion in 2016. The steady increase in performance of ICs in the recent past has been mostly supported by an exponential growth in the density of transistors while inversely reducing the transistors' cost, as described by Moore's law. Even though it is still valid today, its end, as such an exponential law "can't continue forever" was already preconized by Moore itself, and is pushing for new technologic advancements outside complementary metal-oxide-semiconductor (CMOS) IC design. In the meanwhile, telecommunications, medical, and multimedia applications extensively use of electronic devices where blocks of analog and mixed-signal (AMS), digital processors and memory blocks are integrated together. While AMS components in these system-on-a-chip (SoC) designs usually occupy approximately 20 % of the die area, the design effort and respective design costs are considerably higher in comparison to their digital counterpart. In this context, the lack of mature automation tools for analog design has its share of blame in this SoC design paradigm.

Generally the complexity in the design of analog circuits is not due to the number of devices, but from their sensitivity to noise and the countless interactions between them, e.g., parasitic disturbances, crosstalk, substrate noise, thermal noise, etc. Plus, for smaller technology nodes with the increasing complexity of design rules and physical effects, the impact of these interactions is even greater. Despite the algorithms and techniques introduced in the last 25 years, analog IC design automation tools still strive to keep up with the new challenges created by technological evolution. Therefore, designers' exploration of the solution space keeps on being mostly manual, and their knowledge and experience crucial in making effective decisions at all stages of the analog design flow, creating additional risks and elongating the time needed to complete the design. On the other hand, in the digital IC design realm several mature electronic design automation (EDA) tools and design methodologies are available, pushing the design productivity forward.

Currently most of the low-level phases of the process are automated that help the designers keeping up with the new capabilities offered by the technology and making design reuse the commonplace. Since this difference in the level of automation becomes critical when digital and analog circuits are integrated together, the rising of efficient and state-of-the-art tools to boost analog designers' productivity is an intensive research topic in both academia and industry.

The work presented in this book belongs to the scientific area of electronic design automation and addresses automatic sizing of analog ICs. The developed multi-objective design methodology for automatic analog IC sizing was implemented in the tool AIDA-C. In AIDA-C, the usage of state-of-the-art multi-objective multi-constraint optimization engines enables the exploration of circuit design trade-offs. Process variation effects on circuit's performance are accounted with user-defined worst-case corners, and circuit's performance evaluation is done with industrial circuit simulators, e.g., Mentor Graphics' ELDO®, Synopsys' HSPICE®, or Cadence's Spectre®, ensuring that the developed automatic circuit sizing is compliant with the accuracy requirements of analog designers. In addition, layout effects are included in the sizing-flow to decrease the number of independent sizing and layout iterations required to obtain a post-layout correct design. To further enhance AIDA-C, a model for first-order interactions between design variables and circuit performance, the Gradient Model, derived using machine learning techniques, is used to guide and accelerate the optimization process. The proposed approach was validated with both classical and new analog circuit structures for a several design processes, i.e., 130-nanometer, 180-nanometer, and 350-nanometer design kits, showing its validity and generality.

This work would not have been possible without the contribution, and the support and valuable discussions on circuits, layout and optimization, of Ricardo Póvoa, António Canelas, Frederico Rocha, and Ricardo Lourenço.

Finally, the authors would like to express their gratitude for the financial support that made this work possible. The work developed in this book was supported in part by the Fundação para a Ciência e a Tecnologia (Grant FCT-SFRH/BPD/104648/2014, Grant FCT-SFRH/BD/86608/2012, Research project DISRUPTIVE EXCL/EEI-ELC/0261/2012 and Research project UID/EEA/50008/2013) and by the Instituto de Telecomunicações (Research project OPERA-PEst-OE/EEI/LA0008/2013).

This book is organized in eight chapters.

Chapter 1 presents a brief introduction to analog IC design automation, with special emphasis to automatic circuit sizing and optimization taking into account layout effects and random variations. The motivation to address automatic analog IC design is given, then, a well-accepted design flow for analog ICs that is the starting point for the methodology proposed in this book is described.

Chapter 2 presents a study of the available tools for analog design automation, overviewing state-of-the-art sizing optimization, robust design and layout-driven approaches. Valuable information can be gathered from them, such as algorithms and evaluation methods.

Chapter 3 introduces the developed automatic flow for analog IC design and particularly the general description of the developed methodology providing more detail about the inputs and interfaces of AIDA-C.

Chapter 4 describes the multi-objective optimization techniques used in AIDA-C, addressing the non-dominated sorting genetic algorithm II (NSGA-II), multi-objective simulated annealing (MOSA) and multi-objective particle swarm optimization (MOPSO) algorithms. The enhancements made by using machine leaning techniques, i.e., the Gradient model, are also explored.

Chapter 5 studies the results obtained with the proposed IC sizing approach with different parameters for the optimization kernel, the impact of considering nominal or worst-case conditions in the evaluation of the circuits, and also, the advantages of enhancing the optimization kernels with the gradient model. Also, two amplifiers and an oscillator are used to compare the performance of the three optimization kernels (NSGA-II, MOSA MOPSO).

Chapter 6 presents two new methodologies to include layout effects in the sizing optimization loop: the floorplan-aware approach, which is a method to include layout's geometric properties in the optimization with negligible impact in the performance; and the layout-aware approach that accounts for the parasitic effects.

Chapter 7 presents the results obtained with the proposed analog layout-aware sizing approach, by considering the circuit's floorplan and layout induced parasitics. The efficiency of the methodology is proved in the successful design and area improvement of four design cases: a single-stage folded cascode amplifier with bias, a single-stage amplifier with gain enhancement using voltage combiners, a two-stage Miller amplifier, and a two-stage folded cascode amplifier, for a 130-nanometer design process.

In Chap. 8, the closing remarks and future directions for the continuous development of AIDA-C are outlined.

Lisbon, Portugal

Nuno Lourenço
Ricardo Martins
Nuno Horta

Contents

Abbreviations

AIDA	Analog IC Design Automation
AMG	Analog Module Generator
AMOSA	Archive-based Multi-Objective Simulated Annealing
AMS	Analog- and/or Mixed-Signal
CAD	Computer-Aided Design
CMOS	Complementary Metal-Oxide-Semiconductor
DOE	Design of Experiments
DRC	Design-Rule Check
EDA	Electronic Design Automation
EM	Electromigration
FOM	Figure Of Merit
GA	Genetic Algorithm
GDS	Graphic Database System
GUI	Graphical User Interface
HDL	Hardware Description Language
IC	Integrated Circuit
LDE	Layout Dependent Effect
LDS	Layout Description Script
LHS	Latin Hypercube Sampling
LNA	Low-Noise Amplifier
LP	Linear Programming
LVS	Layout-Versus-Schematic
MC	Monte Carlo
MIM	Metal-Insulator-Metal
MOEA	Multi-Objective Evolutionary Algorithm
MOM	Metal-Oxide-Metal
MOO	Multi-Objective Optimization
MOPSO	Multi-Objective Particle Swarm Optimization
MOSA	Multi-Objective Simulated Annealing
NSGA	Non-dominated Sorting Genetic Algorithm

OTA	Operational Transcondutance Amplifier
POF	Pareto Optimal Front
PVT	Process, Voltage and Temperature
RF	Radio Frequency
SA	Simulated Annealing
SO	Single-Objective
SoC	System-on-a-Chip
SVM	Support Vector Machine
UMC	United Microelectronics Corporation
VCO	Voltage Controlled Oscillator
VLSI	Very Large Scale Integration
XML	Extensible Markup Language

List of Figures

List of Tables

Chapter 1
Introduction

In the last decades very large scale integration technologies have been widely improved, allowing the proliferation of consumer electronics and enabling the steady growth of the IC market from $10 billion in 1980 to an estimated value of over than $350 billion in 2016 [1]. The steady increase in performance of ICs in the recent past has been mostly supported by an exponential growth in the density of transistors while inversely reducing the transistors' cost, as described by Moore's law [2]. Moore's law is a law of economics not physics, and Moore itself already preconized its end, as such an exponential law "can't continue forever" and is pushing for new technologic advancements outside complementary metal-oxide-semiconductor (CMOS) IC design, but it is still valid today.

In this context, the increasing need of faster, optimized and reliable electronic devices urges their cost-effective development to efficiently meet customers' demand under highly competitive time-to-market pressure [3]. Moreover, the design of such complex multimillion transistor ICs is only possible because designers are assisted by computer aided design (CAD) and design automation tools that support the design process.

1.1 Analog IC Design Automation

As analog ICs are difficult to design and reuse, designers have been replacing analog circuits by digital computing whenever possible. Hence, most of the high-level functions are implemented using digital or digital signal processing, however analog and radio frequency (RF) circuitry is needed to interface with the real world, and some functionalities that are intrinsically analog [3, 4] are listed below:

- **Sensing the system inputs**: The signals of a sensor, microphone or antenna has to be detected or received, amplified and filtered, to enable digitalization with good signal-to-noise and distortion ratio. Typical applications of these circuits are in sensor interfaces, telecommunication receivers or sound recording;

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0_1

- **Converting analog signals to digital signals**: Mixed-signal circuits such as sample-and-hold, analog-to-digital converters, phase-locked loops and frequency synthesizers provide the interface between the input/output of a system and digital processing parts of a system-on-chip (SoC);
- **Converting the digital output back to analog**: The signal from digital processing must be converted and strengthened to analog so the signal can be conducted to the output with low distortion;
- **Provide and regulate power**: Voltage/current reference circuits and crystal oscillators offer stable and absolute references for the sample-and-hold, analog-to-digital converters, phase-locked loops and frequency synthesizers;
- **The implementation at transistor level of the digital gates**: The last kind of analog circuits are the extremely high performance digital circuits. As exemplified by microprocessors that are custom sized, as analog circuits, to achieve highest speed and lowest power consumption.

Telecommunications, medical and multimedia applications make extensive use of electronic devices where blocks of analog and mixed-signal (AMS), digital processors and memory blocks are integrated together [5]. The growth in Medical/Health, automotive, LED lighting, and energy management for buildings is likely to keep analog needs growth. In 1980, analog ICs represented 32 % of total IC shipments, that value increased to 49 % in 2010, and is forecast to grow to 57 % of total IC shipments by 2018 [6]. In this context, the development and improvement of CAD tools that increase analog designers' productivity is an urgent need.

In the digital IC design several mature electronic design automation (EDA) tools and design methodologies are available that help the designers keeping up with the new capabilities offered by the technology, and, making circuit reuse usual, leading to an increased design productivity. Currently almost all low-level phases of the process are automated. The system is described in a hardware description language (HDL) such as VHDL or Verilog, either at the behavioral level or at the structural level. High-level synthesis tools attempt to synthesize the behavioral HDL description into a structural representation. Logic synthesis tools then translate the structural HDL specification into a gate-level netlist, and semi-custom layout tools (place and route) map this netlist into a correct-by-construction mask-level layout based on a cell library specific for the target technology process.

Research interest is moving in the direction of system synthesis where a system-level specification is translated into hardware–software co-architecture with high-level specifications for hardware, software and interfaces. Reuse methodologies and platform-based design methodologies are also being developed to reduce even further the design effort for complex systems. The level of automation is far from the push-button stage, but the developments are keeping up reasonably well with the circuits' complexity supported by the technology [3, 4].

On the other hand, and despite the algorithms and techniques introduced in the last 25 years, analog IC design automation tools strive to keep up with the new challenges created by technological evolution [4, 7–9]. Designers' exploration of the solution space is mostly manual, elongating the time needed to complete the

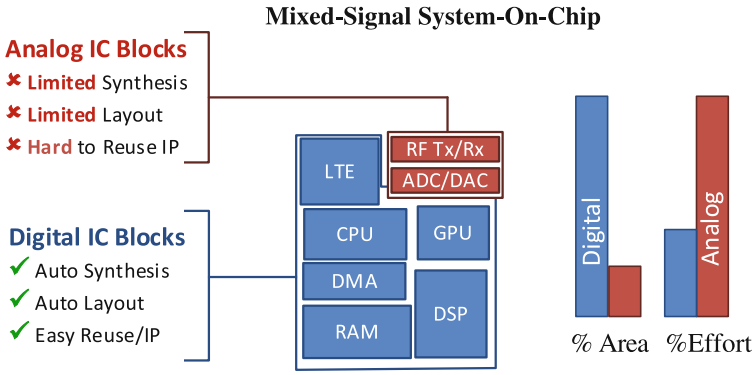


Fig. 1.1 Digital versus analog design automation reality [10]

design task. Generally the complexity of designing analog circuits' is not due to the number of devices, but from the countless interactions between them. Plus, for smaller technology nodes with the increasing complexity of design rules and physical effects, the impact of these interactions is even greater.

Due to the nature of the signals handled, analog circuits are more sensitive to parasitic disturbances, crosstalk, substrate noise, supply noise, thermal noise, etc., and the variety of schematics and diversity of devices' sizes and shapes are much larger [4, 8]. The result of this automation deficit is that, while analog parts in a SoC occupy only approximately 20 % of the global circuit area, as shown in Fig. 1.1, the design effort is considerably higher in comparison to the design effort of the digital section.

This difference in the level of automation between analog and digital design is because analog is, in general, less systematic, more heuristic and knowledge intensive than the digital counterpart, and becomes critic when digital and analog circuits are integrated together. As the analog automation tools do not progress at the same pace of technology, knowledge and experience of the designer is always crucial for making decisions at all stages of the analog design flow.

In short, the development time of analog blocks is considerably higher when compared to the development time of the digital one. The three main reasons for the larger development time are: there is a lack of effective EDA tools; analog circuits are being integrated using technologies optimized for digital circuits; and, analog blocks are difficult to reuse because they are more sensitive to surrounding circuitry, environmental and process variations than their digital counterpart [3, 4].

1.1.1 Robustness in Analog IC Design

As mentioned, despite the advantages that new fabrication technologies bring to systems' performance, the huge increase in device density did not came only with

benefits. This is especially true for analog circuits. The reduced size and high density of devices in modern ICs add new challenges to analog designers, as the effects of non-idealities and variability of the fabrication process parameters in both space and time became more significant. This phenomenon affects devices in different chips but also devices within the same chip, and is solved by robust circuit design with several compensatory techniques [11] to ensure that the vast majority of the fabricated circuits will work according to specifications.

The most common techniques for analog design centering are Monte Carlo Simulation, process, voltage and temperature (PVT) Corners and Worst Case Corners. Monte Carlo simulation executes many simulations by applying random variations to circuit's and process' parameters, making a stochastic sampling of the behaviors of the circuit in real world conditions. PVT Corner is a worst-case approach where the circuit is simulated over multiple combinations of extreme process parameters, voltage supply and temperature variations. Another kind of worst case approach is the worst case corners, which are the set of environment and process parameters that lead to the worst case value for performance figures. Figure 1.2 illustrates 8 corners cases obtained by considering 3 values for power supply, operating temperature and process parameters, together with a set of Monte Carlo samples around the nominal performance for performance figures P_1 and P_2, and the worst case corners for each performance (higher is better) [12].

Layout-induced parasitic effects are also more and more significant, having the potential to affect noticeably the performance of the sized circuit. To overcome the increasing impact of layout parasitic effects in circuit's performance, sizing and layout design phases tend to overlap. The integration of layout generation or layout-related data in the sizing optimization process helps to account for the effects of such non-idealities and parasitic disturbances earlier in the design flow, reducing the number of iterations required to achieve designs that meet post-layout specifications. It is also important to note that careful layout considerations, like symmetry, matching, isolation, proximity, thermal balancing, etc., really help to mitigate some of the problems introduced by the spatial variability [13–15].

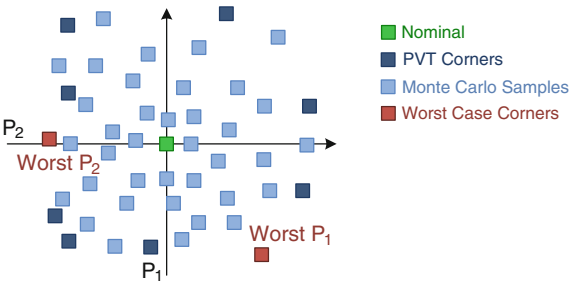


Fig. 1.2 Common methods to measure the effects of variation around the nominal circuits' performance

1.1.2 Computer Assisted Analog IC Design

The systems' complexity and the extremely competitive markets obligate the use of CAD tools to enhance the design process. Today's analog design environment is made of CAD tools for analog IC design editing, evaluation and verification that are mature and fully deployed in the industrial environment, the following list some examples:

- Simulation tools: ADiT, Questa and Eldo [16]; HSPICE, nanosim and HSim [17]; Spectre [18]; SMASH [19].
- Layout edition: Virtuoso Layout Editor [18]; IC Station Layout [16]; Galaxy Custom Designer LE [17].
- Design rule verification and layout extraction: DIVA and Assura [18]; Hercules [17]; CALIBRE [16].
- Integrated platforms like Mentor Graphics IC design Manager [16] or Cadence Virtuoso Analog Design Environment [18].

The time required to manually implement an analog project is usually of weeks or months, which is in opposition to the market pressure to accelerate the release of new and high performance ICs. To address all these difficulties and solve the problems, EDA tools are a solution increasingly strong and solid. Some commercial automation solutions began to emerge as the result of the research efforts in this field, such as ADA's Genius product line and Magma Titan that were integrated in Synopsys (2004 and 2012 respectively) [17], the circuit optimizer feature of Cadence's Virtuoso Custom Design Platform GXL [18], Solido's Fast PVT, Fast Monte Carlo and High-Sigma Monte Carlo [20] or MunEDA's WiCkeD™ [21]. Even if applicable only at cell level, analog components with 10–100 devices [8], they increase the automation level of the analog design environment.

Despite its fundamental aid to designers, the automation options are limited and the ones available are not usually used by the majority of the designers. These tools should play a key role in analog IC design productivity, as they speed up the design process and increase tractability. However, the lack of mature and robust automation tools still leads to handcrafted design, and analog design automation is still a topic of intensive research effort.

1.2 Analog IC Design Flow

Before proceeding into analog IC circuit design automation details, a brief overview on the analog design flow is first provided. Due to the nature of analog design, it is difficult to identify one unique design flow. Nevertheless, a typical and well accepted design flow for AMS ICs was proposed by Gielen and Rutenbar in [4]. It consists of a series of top-down topology selection and circuit sizing steps and bottom-up layout generation and extraction steps. Before passing between any of

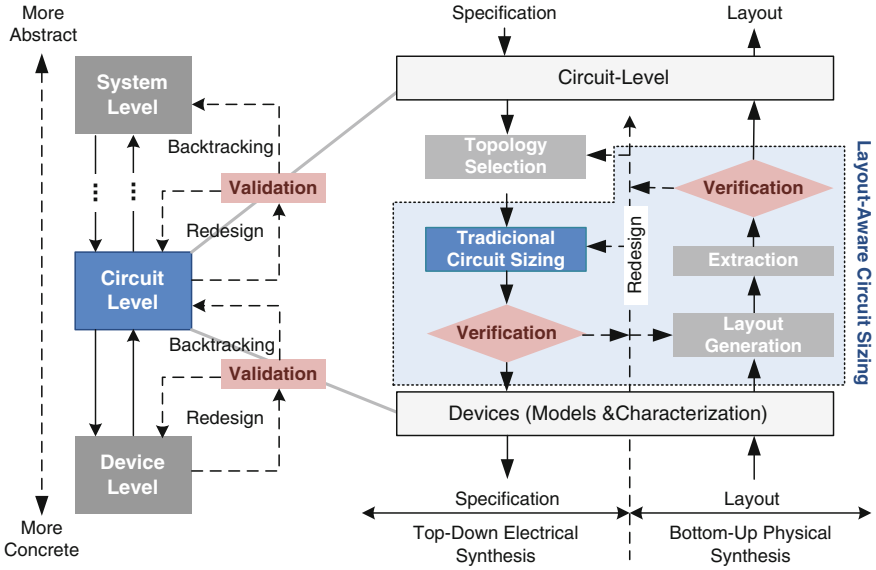


Fig. 1.3 Hierarchical level and design tasks of design flow architectures

the hierarchy levels, in both top-down and bottom-up path, verification is performed to ensure the specifications are met. The design flow is illustrated in Fig. 1.3, considering only system-level and cell-level without the loss of generality.

On the top-down path, the Topology Selection is the process where a set of blocks and the connections between them in defined in order to implement the input specifications of the current hierarchy level. In the Circuit Sizing task, the higher-level specifications are translated in the specifications for each of the blocks. Block specifications may be the definition of the DC Gain and Bandwidth for an amplifier, or the sizes of the transistors, depending on the models used in that abstraction level. The sizing is then verified to ensure the fulfillment of the input specifications. Then, the specifications for each block are passed to the next level of the hierarchy, and the process is repeated until the layout of the innermost block is done. At this point, the bottom-up flow is executed. First, the layout of the current level is generated from the layouts of the deeper hierarchy, then, the layout is extracted to a model suitable for verification. When the top-most level verification is complete, the system is designed. It is important to note that any of the many verification stages throughout the design cycle may detect potential problems, with the design failing to meet the target specifications. In that case, backtracking and redesign will be needed.

The number of hierarchy levels depends on the complexity of the system being handled and there are no generally accepted representations. The same authors in [22] defined the cell-level and system-level, where the cell-level was defined as analog components with 10–100 devices, examples of such circuits are OTAs,

LNAs, Oscillators, Filters, etc., and the analog system-level as analog components with possibly hundreds of these systems, examples of such systems are data converters, phased lock loops, RF front ends, etc. More hierarchical levels can be included in the design flow for complex systems in order to have design tractability. In such intermediary levels the strategies for either system-level or cell-level design automation are applicable, where the most suitable strategy for each level depends on the depth in the design hierarchy and the complexity of the components handled.

At circuit-level, a design task is performed for each analog block. This process is executed iteratively in order to determine the physical dimensions of each device. In this phase two major tasks are considered: the selection of circuit topology and the circuit sizing. The specifications are then, verified through simulation of the circuit by, e.g., using HSPICE® or Spectre®. After the analog blocks have been sized, the project enters into the next phase where the analog blocks are mapped into a physical representation of the circuit, the layout. The layout is a set of geometric shapes that obey to the rules defined by the manufacturing process. Then, the layout passes the design rule check and the layout-versus-schematic verification stages, and finally, it is extracted and simulated to verify the impact of layout parasitic effects on the overall performance of the circuit.

In a traditional analog design, the designer defines a methodology and interacts manually with a set of separate tools in order to achieve the project objectives, aiming to obtain a robust design at the end. Specifically, in the circuit sizing task, the target is to obtain best sizes of the devices, such that the circuit meet the desired specifications, and possibly optimize some of the application specific performance figure (DC Gain, power, area, etc.), However, these objective functions, which relate performance specifications of the circuit with the sizes of the devices, are characterized by being complex, multidimensional and irregular, making the manual search for the ideal solution difficult to achieve. In the manual design, the designers start by finding an approximate solution using simplified analytical expressions, and then, iteratively, adjust the solution until it meets all specifications, which sometimes can be very time consuming.

Verification is done using circuit simulations that provide extra accuracy to the simplified (but treatable) equations used to derive the initial solution. The designer's experience and knowledge are of the utmost importance, as they allow simplifications that speed up the design process, without compromising the quality of the solution.

1.3 Automatic Analog Circuit-Level Sizing

The focus of this book is automatic circuit-level sizing and optimization, where from the set of specifications, the tool finds out the sizes for the components, e.g., widths and lengths of the transistors, resistors, capacitors, etc. Particularly, it covers

the sizing of devices in analog ICs using state-of-the-art and innovative multi-objective optimization techniques that include layout effects in the optimization loop with an additional focus on the integration in the designers' flow.

To summarize these introductory paragraphs, the precise issues addressed in this work are:

- **To allow the analysis of optimal design tradeoffs**: The multi-objective optimization enables accurate design tradeoff analysis; therefore the goal is to provide accurate insights on the optimal tradeoffs that facilitate the analysis of the system-level impact of design choices.
- **To generate robust solutions**: Layout-induced effects are to be included in the sizing process and to ensure correct results, circuit's performances are to be evaluated accurately. In addition, some method to handle process and environment variations must be considered.
- **To be easy to use**: One of the main issues with analog design automation tools is that the analog designer does not use them. The developed tool, AIDA, is designed to appear in the design flow in a way that it helps the designer, while taking advantage of his(her) knowledge.
- **To increase design reusability**: An automated design flow should simplify the retargeting to different specifications of an already setup design or to increase the quality of a previously sized solution.

This is the scenario that lead to the development of the AIDAsoft Framework [23, 24] which implements a design flow from a circuit-level (spice netlist) specification to a physical layout (GDSII stream) description, where AIDA-C is the tool for analog IC sizing and optimization and AIDA-L a companion tool for layout generation. More details about the AIDA project at ICG-IT can be found in 'www.aidasoft.com'.

1.4 Contributions to the State-of-the-Art

The main innovative contributions of the developed methodologies proposed in this book can be summarized as follows:

- **Multi-Objective Evolutionary Sizing of Analog ICs with Corners Validation**: General and efficient multi-objective design methodology and tool for automatic analog IC sizing, which takes into account the effects of process variations. By varying the technological and environmental parameters, the robustness of the solutions is enhanced. The effects of corner cases on the Pareto optimal front were investigated together with the tuning of NSGA-II parameters for application to analog circuit sizing. Three different design strategies considering nominal and PVT corners were tested in a benchmark circuit, showing

the effectiveness of multi-objective design of analog cells [25]. The design of analog circuit involves balancing contradictory objectives. Yet, these tradeoffs, which are commonly used for the definition of Figure-Of-Merits (FOMs), are seldom explored. AIDA-C was used in true automatic multi-objective sizing of an LC-VCO where state-of-the-art FOMs are obtained without being explicitly optimized, where the optimal tradeoffs are shown to the designer [26].

- **A New Meta-heuristic Combining Gradient Models with NSGA-II to Enhance Analog IC Sizing**: Presents an innovative approach to enhance AIDA-C, by embedding statistical knowledge from an automatically generated gradient model into the NSGA-II operators. The approach was validated with typical analog circuit structures, using the United Microelectronics Corporation (UMC) 0.13 μm integration technology, showing that, by enhancing the optimization kernel with the gradient model, the optimal solutions are achieved, considerably faster and with identical or superior accuracy [27–29].
- **Explore Rival Multi-Objective Multi-Constraint Optimization Kernels for the sizing of analog ICs**: Three multi-objective multi-constraint optimization, namely, an evolutionary approach with NSGA-II, a swarm intelligence approach with MOPSO and stochastic hill climbing approach with MOSA, are experimented in the sizing of an LC-Voltage Controlled Oscillator and an LC-Oscillator for a 130 nm technology node, and are compared in terms of performance, using statistical results obtained from multiple runs for each one of the oscillators showing that NSGA-II outperforms the other two reference approaches by having a better convergence time, a widespread set of solutions, and, in general, achieving better Pareto fronts [30].
- **Layout-aware Sizing of Analog ICs using Fast Parasitic Estimates**: Unified fully automated circuit-level sizing implemented with a multi-objective multi-constraint optimization together with a design-rule check proved fully automated layout generation based on high level layout guidelines, described in technology independent layout guides. Evolutionary computation techniques are extensively used, at both circuit-level and physical-level, as tool to solve design optimization problems [23, 24]. Complete layout-related data for both circuit's geometric requirements, which are obtained from the real-time in-loop floorplan packing [31], and circuits' electrical performance, which is obtained from the circuit simulator. In order to boost the parasitic extraction efficiency, the need for expensive detailed layout generation, as found in previous state-of-the-art layout-aware sizing approaches, is here circumvented. However, interconnect parasitic capacitances that are major contributors to performance degradation and on-die signal integrity problems, must be accounted for. Therefore, an empirical-based parasitic extraction is performed on a semi-complete layout obtained from the floorplan with only AIDA-L's global routing. The methodology is demonstrated for the UMC 130 nm design process using well-known analog building blocks proving the generality, accuracy and fast execution of the proposed approach [32].

1.5 Conclusion

Even with the progress in analog IC design automation tools, reception by IC design teams is still far from widespread, leading to hard-to-reuse and costly analog designs. While the challenges in analog IC design introduced by the latest fabrication process are pushing analog designers to use automation solutions, there is still a large room for improvement in analog automation solutions. In this chapter some key issues that still need addressing in analog IC design automation are identified and the AIDAsoft framework's features proposed to handle those issues, which will be further described in detail throughout this book, are summarized.

References

1. World Semiconductor Trade Statistics (2014) WSTS Semiconductor Market Forecast Spring 2014. https://www.wsts.org/PRESS/PRESS-ARCHIVE/WSTS-Semiconductor-Market-Forecast-Spring-2014. Accessed 12 April 2016
2. Moore Gordon E (1998) Cramming more components onto integrated circuits. Proc IEEE 86 (1):82–85
3. International Technology Roadmap for Semiconductors (2012) ITRS 2012 Update. http://www.itrs.net/Links/2012ITRS/Home2012.htm. Accessed 18 Dec 2014
4. Gielen G, Rutenbar R (2000) Computer-aided design of analog and mixed-signal integrated circuits. Proc IEEE 88(12):1825–1852
5. Gielen G (2005) CAD tools for embedded analogue circuits in mixed-signal integrated systems on chip. IEE Proc Comput Digital Tech 152(3):317–332
6. IC Insights, Inc. (2014) Analog unit shipments outpacing growth of All IC Product segments. http://www.icinsights.com/data/articles/documents/705.pdf. Accessed 18 July 2014
7. Rutenbar R, Cohn J (2000) Layout tools for analog ICs and mixed-signal SoCs: a survey. In: Proceedings of the 2000 international symposium on Physical design (ISPD '00), San Diego, 2000
8. Gielen G, Maricau E, De Wit P (2010) Design automation towards reliable analog integrated circuits. In: IEEE/ACM international conference on computer-aided design, San Jose, 2010
9. Graeb HE (ed) (2010) Analog layout synthesis. Springer, Berlin
10. Barros MFM, Guilherme JMC, Horta NCG (2010) Analog circuits and systems optimization based on evolutionary computation techniques. Springer, Berlin
11. Maricau E, Gielen G (2011) Computer-aided analog circuit design for reliability in nanometer CMOS. IEEE J Emerg Selected Topics Circuits Syst 1(1):50–58
12. McConaghy T, Breen K, Dyck J, Gupta A (2013) Variation-aware design of custom integrated circuits: a hands-on field guide. Springer-Verlag, New York
13. Pelgrom MJM, Duinmaijer ACJ, Welbers APG (1989) Matching properties of MOS transistors. IEEE J Solid-State Circuits 24(5):1433–1439
14. Pehl M, Zwerger M, Graeb H (2011) Variability-aware automated sizing of analog circuits considering discrete design parameters. In: International symposium on integrated circuits, Singapore, 12–14 Dec 2011
15. Gielen G, Maricau E, De Wit P (2012) Designing reliable analog circuits in an unreliable world. In: Proceedings of the IEEE 2012 custom integrated circuits conference, San Jose, 9–12 Sept 2012
16. Mentor Graphics (2014) Mentor Graphics. http://www.mentor.com. Accessed 23 Nov 2014
17. Synopsys (2014) Synopsys. http://www.synopsys.com/. Accessed 17 Oct 2014

18. Cadence (2014) Cadence. http://www.cadence.com/. Accessed 17 Oct 2014
19. Dolphin Integration (2016) Dolphin Integration. http://www.dolphin.fr/. Accessed 22 Dec 2016
20. Solido design automation (2015) Solido design automation http://www.solidodesign.com/. Accessed 12 Apr 2015
21. MunEda (2015) MunEDA. http://www.muneda.com/. Accessed 12 Apr 2015
22. Rutenbar RA, Gielen GGE, Roychowdhury J (2007) Hierarchical modeling, optimization, and synthesis for system-level analog and RF designs. Proc IEEE 97(3):640–669
23. Martins R, Lourenço N, Rodrigues S, Guilherme J, Horta N (2012) AIDA: automated analog IC design flow from circuit level to layout. In: International conference on synthesis, modeling, analysis and simulation methods and applications to circuit design (SMACD), Seville, 19–21 Sept 2012
24. Martins R, Lourenco N, Canelas A, Povoa R, Horta N (2015) AIDA: Robust layout-aware synthesis of analog ICs including sizing and layout generation. In: International conference on synthesis, modeling, analysis and simulation methods and applications to circuit design (SMACD), Istanbul, 7–9 Sept 2015
25. Lourenço N, Horta N (2012) GENOM-POF: multi-objective evolutionary synthesis of analog ICs with corners validation. in: genetic and evolutionary computation conference, Philadelphia, 2012
26. Póvoa R, Lourenço R, Lourenço N, Canelas A, Martins R, Horta N (2014) LC-VCO automatic synthesis using multi-objective evolutionary techniques. In: 2014 IEEE international symposium on circuits and systems (ISCAS), Melbourne VIC, 2014
27. Rocha F, Martins R, Lourenço N, Horta N (2014) Electronic design automation of analog ICs combining gradient models with multi-objective evolutionary algorithms. Springer International Publishing
28. Rocha F, Lourenço N, Póvoa R, Martins R, Horta N (2014) A new Metaheuristic combining gradient models with NSGA-II to enhance analog IC synthesis. In: 2013 IEEE Congress on Evolutionary Computation, Cancun, 2013
29. Rocha F, Martins R, Lourenço N, Horta N (2013) Enhancing a layout-aware synthesis methodology for analog ICs by embedding statistical knowledge into the evolutionary optimization kernel. In: Camarinha-Matos L, Tomic S, Graça P (eds) Technological innovation for the internet of things. 4th IFIP WG 5.5/SOCOLNET Doctoral Conference on computing, electrical and industrial systems, DoCEIS 2013, Costa de Caparica, Portugal, April 15–17, 2013. Proceedings, IFIP advances in information and communication technology, vol 394. Springer, Berlin, Heidelberg
30. Póvoa R, Lourenço R, Lourenço N, Canelas A, Martins R, Horta N (2015) Synthesis of LC-oscillators using rival multi-objective/multi-constraint optimization Kernels. In: Fakhfakh M, Tlelo-Cuautle E, Fino M (eds) Performance optimization techniques in analog, mixed-signal, and radio-frequency circuit design. IGI Global, Hershey PA, pp 1–27
31. Lourenço N, Canelas A, Póvoa R, Martins R, Horta N (2015) Floorplan-aware analog IC sizing and optimization based on topological constraints. Integration, VLSI J Elsevier 48:183–197
32. Lourenco N, Martins R, Horta N (2015) Layout-aware sizing of analog ICs using floorplan and routing estimates for parasitic extraction. In: 2015 design, automation and test in europe conference and exhibition (DATE), Grenoble, 9–13 March 2015

Chapter 2
Previous Works on Automatic Analog IC Sizing

2.1 Analog IC Sizing Automation: An Historical Perspective

Historically, automatic analog IC sizing has been implemented using two different approaches, namely knowledge-based and optimization-based approaches [1], illustrated in Fig. 2.1. Knowledge-based approaches systematize the design using a design plan derived from expert knowledge. In these early strategies, a plan is built with design equations and a design strategy to produce the device sizes that meet the performance requirements. In [2], the designer expertise is captured in a design plan where all design equations are explicitly solved during the definition of the plan. Once the topology is selected, the plan is executed for the given specifications and produces a first design. In [3] the same overall strategy is used, but defines the circuits hierarchically, with a design plan for each sub-block. It also adds back-tracking with design-reuse methodologies to recover from failed designs. A slightly different approach is found in [4–7], where designer's knowledge captured in expert systems using artificial intelligence techniques. The main advantage of knowledge-based approaches is the short execution time. However, deriving the plan is hard and time-consuming, it requires constant maintenance to keep it up to date with technological evolution, and also, the results are not optimal, suitable only as a first-cut-design.

Aiming for optimality, the next generations of sizing tools apply optimization techniques to analog IC sizing. Optimization-based sizing tools come in many flavors, but they can be generally classified into two main subclasses: equation-based and simulation-based circuit optimization, from the bulk methods used to evaluate circuit's performance. Equation-based methods use analytic design equations, whereas simulation-based use a circuit simulator to relate the circuit's performance figures with the design variables. While the first take advantage of fast execution time, the development and maintenance of the equations are costly, in addition the lack of accuracy in the evaluation of the circuits' performance greatly

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0_2

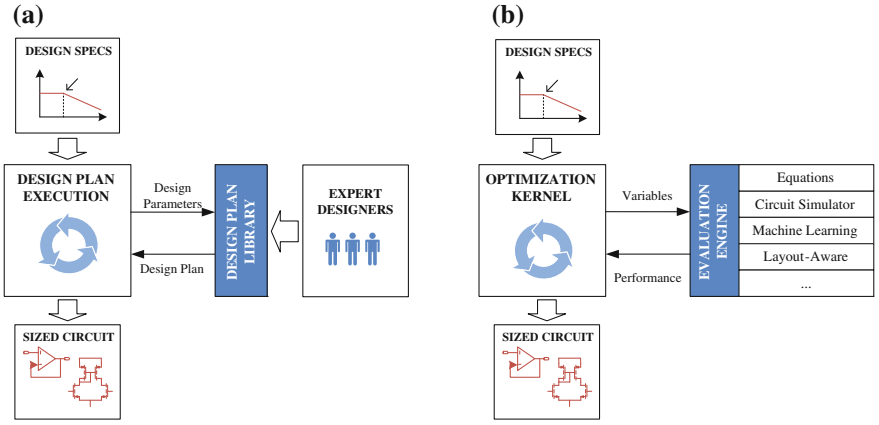


Fig. 2.1 Automatic circuit sizing approaches. **a** Knowledge-based. **b** Optimization-based

penalizes this approach. To tackle the accuracy issue, simulation-based evaluation takes advantage of the circuit simulator, e.g., SPICE [8], to provide accurate performance figures. The cost of this approach is obviously execution time, but with the computation resources available today, quasi-optimal solutions can be obtained with acceptable execution timeframes [9].

In the current state-of-the-art, optimization-based is the most common approach to automatic sizing at circuit level, and some flavors of it are already integrated and available in the toolkits provided by most EDA providers, like the circuit optimizer feature of Cadence's Virtuoso Custom Design Platform GXL [10], Synopsys Titan ADX [11] or MunEDA-GNO [12]. Nevertheless, for system level, design plans or knowledge-based tool are still around [9]. The following sections will overview the most preeminent methods in analog integrated circuit sizing and the latest developments in the area.

2.2 Optimization-Based Circuit Sizing

The critical points in optimization-based approaches are the optimization techniques employed and the methods used to evaluate the circuit's performance. Sect. 2.2.1 overviews the first and Sect. 2.2.2 the second.

2.2.1 Optimization Techniques Applied to Analog Circuit Sizing

Different optimization techniques are used for analog IC sizing and optimization, both deterministic and stochastic. In equation-based methods, as the analytic

equations that relate the circuit's performance figures to the design variables are known, is possible to use classical optimization methods. By using the circuit simulator, methods that take advantage of some properties of the models, e.g., quadric or geometric programming cannot be used, leading, to stochastic heuristic optimization techniques.

Early approaches to simulation-based automatic sizing used local optimization around a "good" solution, where simulated annealing (SA) [13] was the most common optimization technique used. SA mimics the annealing of material under slow cooling to minimize the internal energy, as the name suggests. In DELIGTH. SPICE the optimization algorithm (phase I-II-III method of feasible directions) is used to perform local optimization around a user provided starting point. Kuo-Hsuan et al. [14] and FASY [15, 16] use equations to derive an approximate solution, and then use SA to optimize the design. Likewise, Cheng et al. [17] also uses SA with a good starting point, but considers the transistor bias conditions to constrain the problem. FRIDGE [18] aims for general applicability approach by using an annealing-like optimization without any restriction to the starting point, whereas Castro-Lopez et al. [19] use SA followed by a deterministic method for fine-tuning to perform the optimization.

Other widely used class of optimization methods are genetic algorithms (GAs) [20]. The principle of the GA is to mimic natural evolution where new solution vectors are obtained from the current population by the application of the genetic operators of mutation and crossover. Barros et al. [21, 22] presents a circuit sizing optimization supported by a GA where the evaluations of the populations was made using a both the circuit simulator and a support vector machine (SVM). Alpaydin et al. [23] use hybridization of evolutionary and annealing optimization, where the circuits' performance figures are computed using a blend of equations and simulations. Given the affinity evolutionary algorithms have with parallel implementations, in Santos-Tavares [24], MAELSTROM [25] and ANACONDA [26] the time to simulate the population reduced by a parallel mechanism that shares the evaluation load among multiple computers.

A different approach to circuit sizing optimization that also employs evolutionary methods is to simultaneously generate the circuit topologies (the arrangement of the devices) and the device sizes. Koza [27], Sripramong [28], and more recently Hongying [29] proposed a design methodology able to create new topologies by exploring the immense possibilities starting from low abstraction level. Small elementary blocks are connected bottom-up to each other to form a new topology. Various fundamental entities can be applied, such as, single transistors, elementary building blocks or node connections. However, these approaches are met with great skepticism, as designers are suspicious of the generated structures, because they often differ too much from the well-known analog circuit structures.

Swarm intelligence algorithms [30] can also be found in the literature applied to analog circuit sizing. The fundament of swarm intelligence algorithms is to use many simple agents that lead an intelligent global behavior, like the one observed in many insect hives. From these methods, the most common are the ant colony

optimization (ACO), which was successfully applied in [31, 32], and particle swarm optimization (PSO) that can be found in [33–35].

Circuit sizing is in its essence a multi-objective multi constraint problem, and the designer often explores the tradeoff among contradictory performance measures, for example, minimizing power consumption while maximizing bandwidth, or maximizing gain and minimizing area of an amplifier, therefore, the usage of multi-objective optimization techniques is becoming more common. When considering multiple objectives the output is not one solution, but a set of optimal tradeoff solutions, usually referred as Pareto optimal front (POF). Given the multiple elements already present in both evolutionary and swarm intelligence algorithms, these are the natural candidates to implement such approach. In MOJITO [36] the evolutionary multi-objective methods are applied, respectively, to circuit sizing and both sizing and topology exploration, whereas in [35] the particle swarm optimization is explored in both single and multi-objective approaches. A different approach is taken by Pradhan and Vemuri in [37], where the multi-objective simulated annealing (MOSA) is used.

Instead of executing sizing on-the-fly, in some approaches the non-dominated solutions are generated using the previously mentioned multi-objective optimization methods or variations of them for the most relevant tradeoff (prior to the design task) and then, the suitable solution is selected from the already sized solutions [38–41].

2.2.2 Circuit's Performance Evaluation

As stated, in terms of evaluation methods, the two main subclasses are equation-based and simulation-based evaluation. As both have advantages and disadvantages, hybrid solutions considering both approaches are also common.

2.2.2.1 Equation-Based

In the equation-based approaches different optimization techniques are used. The optimization in OPASYN [42] is done using steepest descent, whereas in STAIC [43], a successive solution refinements technique is used. OPTIMAN [44] uses SA applied to analytical models created automatically by ISAAC [45]. DONALD [46] is an interactive design space exploration tool that assists the designer during circuit sizing by automatic analytical manipulations of the circuit equations. Maulik et al. [47, 48] define the sizing problem as a constrained nonlinear optimization problem using spice models and DC operating point constraints, solving it using sequential quadratic programming. Matsukawa et al. [49] design $\Delta\Sigma$ and pipeline analog to digital converters solving via convex optimization the equations that relate the performance of the converter to the size of the components. In ASTRX/OBLX [50]

a SA optimization is performed on a cost function defined by equations for dc operation point, and small signal asymptotic waveform evaluation (AWE). This evaluation technique is also used in DARWIN [51].

In GPCAD [52] a posynomial circuit model is optimized using geometrical programming (GP), the execution time is in the order of few seconds, but the general application of posynomial models is difficult and the time to derive the model for new circuits is still high. To reduce the long time dispended in model development several automatic techniques were proposed, in [53, 54] Gielen et al. provide a good overview on symbolic analysis applied to analog ICs. However, some design characteristics are still not easy to extract automatically in terms of analytical expressions with sufficient accuracy. Kuo-Hsuan et al. [14] revisited the posynomial modeling recently, surpassing the accuracy issue by introducing an additional step, where local optimization using SA and a circuit simulator is performed. The same strategy is applied in FASY [15, 16] were analytical expressions are solved to generate an initial solution and a simulation-based optimization is performed to fine tune the solution.

The equation-based methods' strong point is the short evaluation time, making them, like the knowledge-based approaches, extremely suited to derive first-cut designs. The main drawback is that, despite the advances in symbolic analysis, not all design characteristics can be easily captured by analytic equations, making the generalization of the method to different circuits very difficult. In addition, the approximations introduced in the equations yield low accuracy designs especially for complex circuits, requiring additional work to ensure that the circuit really meets the specifications.

2.2.2.2 Simulation-Based

With the availability of computing resources, simulation-based optimization gained ground, being the most common approach found in recent approaches. In DELIGTH.SPICE [55] the optimization algorithm (phase I-II-III method of feasible directions) is used to perform local design optimization around a user provided starting point.

FRIDGE [18] on the other hand aims for global optimality by using an annealing-like optimization without any restriction to the starting point. However, to restrict the dimensionality of the problem the user still must provide the range for the optimization variables. In MAELSTROM [25] and ANACONDA [26] the evaluation time is reduced by a parallel mechanism that shares the evaluation load among multiple computer. Given the affinity evolutionary algorithms have with parallel implementations, it was the base technique chosen in MAELSTROM, however and because the success that SA has demonstrated in many implementations, the authors' option was to use parallel re-combinative simulated annealing (PRSA). In ANACONDA the approach is similar, but instead of the PRSA it is

applied a variation of pattern search algorithms, named by the authors as stochastic pattern search.

Generality and easy-and-accurate model (the circuit netlist) are the strong points of simulation-based techniques. However, the execution time is large for complex circuits ($\sim$100 variables) and prohibitive at system level, and, without the proper constraints the algorithm may not converge to a good result. Some heuristic schemes exist to automate the process of defining the constraints [56]. However, automatic constraint defining mechanisms are not integrated in sizing tools and their application is somewhat circuit class specific. Being the high execution time the weaker point of this approach, some techniques had been proposed to cope with it. As stated before, Kuo-Hsuan et al. [14] and FASY [15, 16] use equation-based techniques to derive an approximate solution, and then use simulation within a SA optimization kernel to optimize the design. Cheng et al. [17] use the transistor bias conditions to constrain the problem and instead of solving the circuit by finding transistor sizes, the problem is solved by finding the bias of the transistors. The transistor sizes are then derived from the bias point using electric simulation. In MAELSTROM [25] and ANACONDA [26] the evaluation time is reduced by a parallel mechanism that share the evaluation load among multiple computers.

To handle complex circuit where the simulation is too expensive, [57] takes a hierarchical approach where equations are used for system level evaluation, while the circuit simulator is used for block level. The sizing flow is neither top-down of bottom-up, but concurrent optimization of both system and block level.

In order to increase the performance of simulator-based circuit optimizers, surrogate-based evaluation is maturing as an alternative technique to avoid the costly simulations [58, 59]. In such approaches, the global accuracy of the simulator is traded by the speed of applying the surrogate model for performance estimation. These models are automatically generated which eases design and maintenance, however tuning the model accuracy can be complicated, which makes them hard to use in a generalized form.

Alpaydin et al. [23] use a neural-fuzzy model combined with an evolutionary optimization strategy where some of the AC performance metrics are computed using an equation-based approach. De Bernardinis et al. [60] use a learning tool based in SVM to represent the performance space of analog circuits. The performance space is modeled using the knowledge acquired from a training set via circuit simulation. Wolfe et al. [61] present a performance macro-model based in a neural network. This model once constructed, is used to replace the SPICE simulation during the sizing of analog circuits, increasing the efficiency of the performance parameter estimates' computation. The training and validation data sets are constructed with discrete points, sampled over the design space. The work explores several sampling methodologies to adaptively improve model quality and applies a sizing rules methodology in order to reduce the design space and ensure the correct operation of analog circuits. Barros et al. [21, 22] present a cell-level sizing and optimization approach based on SVMs and evolutionary strategies. The SVM is used to dynamically model performance space and identify the feasible design space regions while at the same time the evolutionary techniques are looking for the

global optimum. The evaluation is still done with HSPICE to ensure accuracy, but the number of evaluation is reduced by using the SVM to prune the candidate solutions, in [62] the approach is generalized for multi-objective.

2.2.3 Commercial Solutions

The EDA industry closely follows the main trends in academia and R&D work-groups, focusing on the lower level of abstraction levels dealing with device sizing and layout description levels. Synopsys titan ADX (originally Magna's) takes an equation-based approach. To overcome the issues with equations, namely the non-generality and the difficulty to derive them, an automated procedure is used to extract the circuit models without user inputs. To ensure accuracy of the models, the automatic procedure uses a circuit simulator. While this automatic procedure solve the problems related to time to design and maintain the models, model accuracy is still an issue, limiting the applicability of the approach to high-performance designs. In the high-end of Cadence products (Virtuoso-GXL), automatic IC optimization is provided with 4 alternative optimization methods that can be used with the simulator. In this approach accuracy is ensured, but the limitative single-objective optimization and the hard-to-use constraint handling together with the long simulation time for real industrial designs, are the main deterrent for generalized use.

2.3 Robust Circuit Optimization

With the evolution of the fabrication processes, new challenges to analog IC design automation arisen, namely the need for circuits that are robust to the variability in both fabrication process and environment. The variability in IC design comes from the environment (voltage, temperature) variations, die-to-die variations that affect all devices identically, and intra-die variations that affect each device differently.

Process variation has been around since the early design processes, but the reiterated reduction in devices' sizes increases the effects of process variation because: transistors may be shrinking, but atoms do not. On large devices some displaced atoms do not cause a significant performance shift, but for small devices the performance shift must be accounted if the vast majority of the design are to meet specifications after fabrication [63].

The environment parameters are usually defined by a lower and upper bound, while the process variation parameters are random variables defined by some probability distribution function (commonly a Gaussian with zero mean). Environment and die-to-die variations are commonly handled using process, voltage and temperature (PVT) corners, while intra-die variations are handled using Monte Carlo (MC) analysis. However, both comprehensive PVT corners and MC

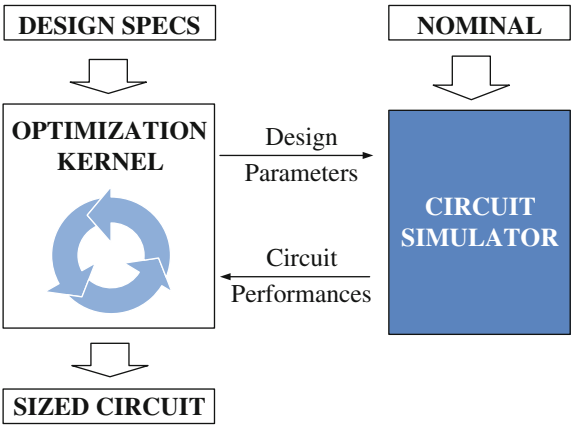


Fig. 2.2 Nominal simulation-based circuit sizing and optimization

require many additional circuit simulations and are inadequate to be used inside the optimization loop.

To circumvent the amount of simulations required, worst case parameters sets or worst case corners are often considered instead [64, 65]. Worst case corners are set of values for the environment and process parameters, which are not controlled by the designer, that lead to the worst values of circuit's performances. However, unlike digital cells where worst case corners do not depend on topology or device sizes and are provided in the models by the foundry (e.g. Slow, Fast, etc.), analog worst case sets do depend of the circuit's topology and sizes of the devices [66]. Figures 2.2 and 2.3 illustrate the differences between nominal and variation-aware circuit sizing and optimization.

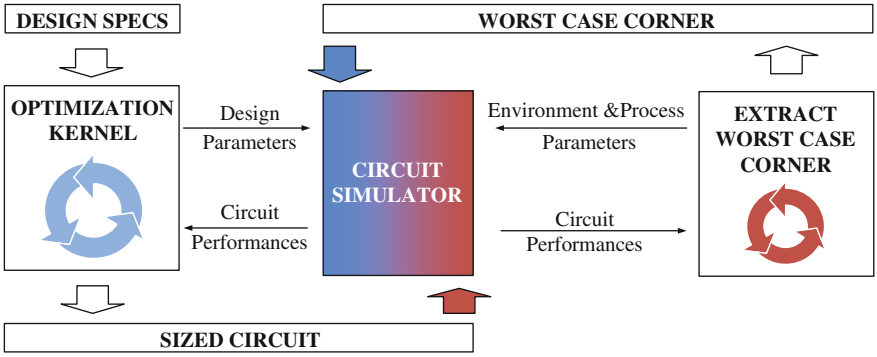


Fig. 2.3 Variation-aware automatic circuit sizing and optimization

2.3.1 Worst-Case Optimization

A simple method to derive these worst case corners, which does not account for intra-die variations, is for the designer to select the relevant worst case corners [21]. A more systematic approach is the "Fast PVT corner" described in [63] that transforms the search for the worst case PVT corner in an explicit optimization problem that minimizes the circuit performance in the PVT corner space. This worst case corner is then considered during the circuit design. As the worst case corner depends of the circuit sizing, regular updates may be needed.

Another approach is to find the worst case parameters sets accounting for intra-die variations. One of such method is the "Sigma driven corners" [63]. The algorithm to extract the parameters that correspond to a parametric yield of $n \times \sigma$ for a given performance target is outlined in Algorithm 2.1. First a MC sampling is done ($\sim$100 points for 3σ), then the p.d.f. of the output is estimated using kernel density estimates. The percentile of the p.d.f. is then used to derive the $n \times \sigma$ performance boundary. The last step is to find the process and environment parameters that cause that performance values.

Algorithm 2.1 Sigma driven corner extraction [63]

1.	Simulate 100 MC samples
2.	Estimate p.d.f. of the circuit's performance
3.	Use the p.d.f. to determine performance bound
4.	Find the corresponding process and environment parameters Starting from the closest point in the sampling

Linear Worst Case distance design [67], assumes that in circuits already designed for nominal conditions, i.e., typical values for temperature and voltage and no process variation, most of the circuits' performance figures can be accurately linearized. It takes a deterministic circuit optimization approach, where the worst case environment and process parameters are iteratively updated using a gradient estimation. Then, at each iteration, the design variables are determined solving the sizing optimization problem considering the latest worst case environment and process parameter set. To increase the accuracy of the estimates, in [68] a quadratic model replaces the linear model. Using the worst case distances concepts, in [69] the bias-driven performance equations are derived to include variability for flexible thin film transistors. A linear-solver is then used to optimize the bias point and look up tables, which were obtained before the sizing task using the circuit simulator to account for all transistor's non-idealities, are used to obtain devices' sizes from their bias.

The above solutions provide a way to handle variability, but they work only over the yield considering each to the circuit's performances separately, and are not adequate for handling the statistical yield (% of fabricated ICs that meet all specifications). On the other side, they provide a direct relation between the variability of each performance with respect to the process and environment variability, that is

extremely useful for the designers to understand which performances figures are critical under variability.

A different approach is to handle yield considering full MC for the relevant solutions and speed the process by avoid useless simulations of less promising solutions. In [70], multiple levels of evaluation accuracy are considered simultaneously. For the initial evaluations only DC simulations are considered, in the next levels other simulations are considered for nominal conditions, followed by PVT corners, and finally, full MC.

This scheme allows global optimization where tentative designs are initially evaluated using simplified (and faster) simulations, and as those tentative solutions get closer to the solution more accurate evaluations are performed. To further accelerate the optimization, a surface response model is generated on the fly, and is used to generate new tentative solutions. A similar approach to efficiently use the computational resources without compromising accuracy is taken in [71], where evolutionary ordinal optimization is used. In ordinal optimization the exact values for yield are not necessary, as long as tentative solutions are well sorted in terms of performance. In this manner, the number of simulations executed to estimate the yield is allocated efficiently to avoid unnecessary yield estimations of bad tentative solutions and, accurately estimate the yield of the best candidates as the execution progresses.

Despite the merit of the previously discussed approaches in handling variability, they are still impractical for high-sigma design. High-sigma design take the amount of simulations required to another level, a 6σ design (approx. 2 error per billion samples) requires billions of MC simulations for verification [63]. Specific methods for high-sigma design have been proposed that reduce the amount of simulations to practical levels [63, 72–74]. Yet, while extremely relevant for bit-cell and other kind of highly redundant blocks that appear thousands of times in an IC, the relevance of high-sigma design is reduced for analog IC design, as most analog cells are not repeated hundreds of times in an IC. Eventually for some stricter applications, 4-sigma design (1 error per 15 thousand samples [63]) might make sense.

In addition to environment and fabrication variability, effects like well proximity effects (WPE), shallow trench isolation (STI) stress effect, electromigration, hot carrier injection (HCI), negative bias temperature instability (NBTI), positive bias temperature instability (PBTI), among others, lead to aging effects that are noticeable in the latest fabrication processes, where the behavior of the devices and consequently the performance of the circuit degrades over time. With respect to this issue, methods to simulate considering aging have been proposed, e.g., in [75] MC circuit simulations (plus a surface model) are used to derive the failure distribution through the lifetime of the circuit, and, in [76], quadratic worst case distances are extended to account both for variation and aging effects.

2.3.2 Commercial Solutions

The growth of the importance of variation effects, created a gap between what traditional tools supported and what the analog designer needed. Taking that opportunity, new commercial EDA solutions emerged to efficiently tackle the variation-aware design. For example, fast PVT corners, sigma-driven corners and high-sigma via sampling ordering are provided as commercial solutions by Solido Design Solutions [77], whereas, MunEDA [11] delivers worst case distances using linearized and quadratic models, and importance sampling, among other solutions.

Much has been achieved in this domain, still, there is a space for innovation in variation-aware circuit sizing, first and foremost, variation-aware design still require an agonizingly high number of circuit simulations to yield accurate results.

2.4 Layout-Aware Sizing

Another source of analog IC performance degradation are the parasitic devices formed in the silicon implementation of the circuit [78]. Hence, the performance of a circuit needs to be guaranteed post-layout in the presence of these layout parasitics, which prevent the circuit from realizing the estimated performance.

In the traditional flow, the layout generation task is only triggered when the sizing task is complete. However, to achieve post-layout successful designs that meet all specifications, time-consuming and non-systematic iterations between these electrical and physical design phases are required. Without them, performance overdesign results in wasted power and area, and if underestimated, the circuits' post-layout performance can be compromised [79]. Furthermore, even the evaluation of the circuit area or aspect ratio is practically impossible from the netlist alone, which is especially critical for designs with geometric constraints, e.g. an upper limit in the layout width, causing expensive redesigns. Thus, to address post-layout performance degradation and geometric requirements earlier in the design flow, the, so called, layout-aware or layout-driven design approaches include layout effects during the sizing loop, as illustrated in Fig. 2.4. Table 2.1 summarizes some state-of-the-art layout-aware sizing tools and their different ways of extracting layout-parasitics, that are presented next.

Without actually generating the layout in real-time, in [37] device parasitic effects are modeled by linear regression from a Pareto optimal surface, obtained by sampling the design space and using a procedural generator to produce the layout for each point, where each solution is aware of its specific layout induced effect. In [80] the layout is also produced by a parameterized generator, where layout parasitics and devices sizes are passed to the precompiled symbolic performance model that estimates the circuit performance for each sizing solution, attempting to avoid circuit simulation. In [81] the parasitic effects are modeled in the circuit equations, a

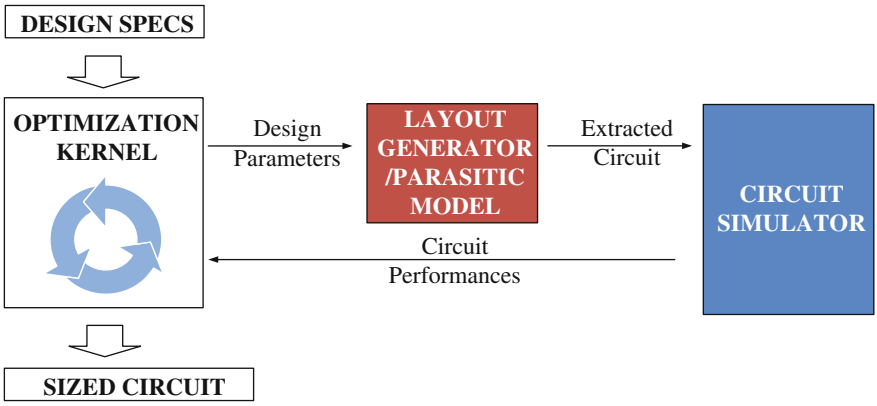


Fig. 2.4 Layout-aware automatic circuit sizing and optimization

template based layout generator is used to derive the parasitic equations that are solve together with the performance equations using non-linear-programming.

In [83] the procedural generator is used during the execution of a design plan, where part of the parasitic effects are estimated using analytical models, and other part is accounted by setting the BSIM's [90] parameters accordingly. Unlike previous approaches, in [82] the procedural generator is used to generate the layout but inside the optimization loop. Likewise in [19, 85] both parasitic-aware and geometrically constrained sizing are handled. Since the predefined layout template is supported by a slicing tree and the block placement is fixed, area and shape optimization are obtained by finding the number of fingers of MOS transistors that yield desired geometric features. In addition, layout parasitic effects are extracted and used during the circuit's evaluation in the optimization loop. This extraction can be done either using analytical equations and layout sampling or using complete 3D geometric extraction models. In [86], a layout description script is also used in-loop in a simulation-based scheme, where the evolutionary strategies algorithm and simulated annealing are used to perform layout-aware optimization of one candidate from the Pareto front.

Finally, in [84] the designer knowledge is embedded in a set of placement and routing matching, symmetry and proximity constraints, as well as geometric and electrical performance constraints. Then, a deterministic nonlinear optimization algorithm is used to determine the device sizes. In each iteration, the floorplan is exhaustively explored by enumeration [91], and then the best placement, selected based on some performance criteria, is routed and considered for post-layout simulation.

Layout-aware/Layout-inclusive sizing tools target a design process that avoids time consuming iterations, by bringing layout-related data into the sizing process. However, both complete automatic layout generation and accurate parasitic extraction are still time consuming and hard to setup operations [79]. The time cost appears either in the processing time, when using custom automatic layout

Table 2.1 Overview of layout-aware sizing tools

Work	Circuit synthesizer	Performance evaluation	Layout generator		Layout data included	
			Placer	Router	Geom.	Parasitics
Vancorenland et al. [82]	Optimization-based	Fitted functions			Eq.	1D/2D critical nets only
Ranjan et al. [79]	Optimization-based	Symbolic models			No	Area interconnect extractor
Pradhan [37]	MOO	Symbolic models			No	Bulk, device and interconnect
Youssef et al. [83]	Design plan to iteratively solve model equations	BSIM4 equations			Yes	Models of devices' stress effects
Habal et al. [84]	Nonlinear deterministic optimizer	Circuit simulator	Enum. of all floorplans	Cadence chip assembly router®	Yes	Commercial extractor
Lopez [19, 85]	MOO	Circuit simulator	Coded slicing-tree	Template-based	Yes	Analytical and commercial extractor
Liao et al. [81]	Nonlinear programming	Bias driven equations	Template based/assisted placement	Channel router	Yes	1D/2D modeling of the critical nets
Berkol et al. [86]	Evolutionary strategies and SA	Circuit simulator	Layout description script		Yes	Commercial extractor
AIDA [87–89]	MOO	Simulator (Spectre®, Eldo® or HSPICE®)	Multiple B*-trees	Automatic electromigration-aware WT and global routing in-loop	Yes	2D/2.5D modeling of the devices and routing

generator plus layout extraction inside the sizing optimization loop [84], or in the design time of circuit specific template-based [19, 85] or procedural generators [37, 79, 82, 83] with parasitic estimation.

2.5 Summary of the Automatic Circuit Sizing Approaches

In this survey, the most significant automatic analog IC sizing tools were presented to provide a better understanding of its advantages and shortcomings. The variety of existing tools and techniques covering several aspects of analog design are summarized in Table 2.2, highlighting the benefits and pitfalls of the main methods to automate analog IC sizing, and the sizing tools surveyed are summarized in Table 2.3.

The following paragraphs frame the most relevant aspects of the simulation-based multi-objective optimization approach developed in this work, faced with the techniques found in the surveyed state-of-the-art methodologies.

Table 2.2 Summary of advantages and shortcomings of the techniques applied to circuit sizing tools

Knowledge-based	Optimization-based		
	Equations	Simulator	Models
IDAC [2] OASYS [3] BLADES [4] CAMP [5] ISAID [6, 7]	OPASYN [42] STAIC [43] Kuo-Hsuan [14] OPTIMAN [44] DONALD [46] ASTRX/OBLX [50] DARWIN [51] GPCAD [52] Doboli [92] Matsukawa [49]	Kuo-Hsuan [14] FASY [15, 16] ASTRX/OBLX [50] DARWIN [51] DELIGTH.SPICE [55] Cheng [17] FRIDGE [18] MAELSTROM [25] ANACONDA [26] Castro-Lopez [19]	Alpaydin [23] De Bernardinis [60] Wolfe [61] Barros [21, 22]
(+) Fast execution time (+) Expert knowledge	(+) Fast execution time (+) Use of expert knowledge (+) Automatic symbolic analysis	(+) Models are easy to develop (++) Accurate and flexible	(+) can be accurate (+) can be flexible
(−) Expert knowledge is difficult to capture (−) Not optimal	(−) Difficult derivation of some equations (−) Simplifications lead to lack of accuracy	(−) Long execution time (−) Limited to cell-level	(−) Accuracy is hard to predict

Table 2.3 Summary of circuit sizing tools

Tool\author	Circuits	Design plan/optim.	Evaluation	1	2	3	Time setup/exec.
IDAC [2]	Analog cells	Design plan plus SA post-optimization	Equations	⃠	✘	✓[6]	months/few sec
DELIGTH.SPICE [55]	Analog cells	Feasible directions optimization	Circuit simulator	✓	✘	✘	moderate/18 h
OASYS [3]	Amplifier	Design plan (includes backtracking features)	Equations	✓	✓[4]	✘	6 months/3 s
BLADES [4]	Amplifier	Expert system for analog design	Equations	⃠	✓[4]	✘	Long/20 min
OPASYN [42]	Amplifier	Steepest descent	Equations	✓	✓[4]	✓[6]	2 weeks/5 min
CAMP [5]	Amplifier	Expert system, flexible architecture	Circuit simulator	⃠	✓[5]	✓[6]	⃠/⃠
OPTIMAN [45]	Amplifier	SA	Analytical models ISAAC [72]	✘	✘	✘	⃠/1 min
SEAS [100]	Amplifier	SA	Equations	✘	✓[5]	✘	⃠/⃠
DONALD [46]	Amplifier	Equation solver (Newton Raphson variant)	Equations	⃠	✘	✘	⃠/⃠
Chang [101]	ADC	Top-Down constraint driven	Behavior models	⃠	✓[5]	✓[6]	⃠/4–89 h
STAIC [43]	Amplifier	2 step optimization	Equations	⃠	✘	✓[6]	long/2 min
MINLP [102]	Amplifier	Branch and bound	Equations and BSIM models	⃠	✓[5]	✘	6 months/1 min
Maulik et al. [47, 48]	Amplifier	Sequential quadratic programming	Equations and BSIM models	⃠	✘	✘	6 months/1 min
FRIDGE [18]	Amplifier	SA	Circuit simulator	✘	✘	✘	1 h/45 min
DARWIN [51]	Amplifier	GA	Small signal, Analytical expressions	✘	✓[5]	✘	⃠/⃠
ISAID [6, 7]	Amplifier	Qualitative reasoning + post optimization	Equations and qualitative reasoning	⃠	✘	✘	⃠/⃠

(continued)

Table 2.3 (continued)

Tool\author	Circuits	Design plan/optim.	Evaluation	1	2	3	Time setup/exec.
SD-OPT [103]	ΣΔ-modulator	SA	Equations and behavioral simulation	✗	✗	✗	Long/1, 5 week
FASY [15, 16]	Amplifier	SA + Gradient	Circuit simulator	✗	✓[4]	✗	⃠/6 h
ASTRX/OBLX [50]	Analog Cells	SA	AWE Equations	✗	✗	✗	few days/few sec
Koza [27]	Analog Cells	GA	Circuit simulator	✗	✓[5]	✗	⃠/⃠
GPCAD [52]	Amplifier	Geometric programming	Posynomial models	✗	✗	✗	⃠/fast
MAELSTROM [25]	Amplifier	GA + SA	Circuit simulator	✗	✗	✗	⃠/3,6 h
ANACONDA [26]	Amplifier	Stochastic pattern search	Circuit simulator	✗	✗	✗	⃠/10 h
Sripramong [28]	Amplifier	GA	Circuit simulator	✗	✓[5]	✗	⃠/3 days
Alpaydin [23]	Amplifier	Evolutionary strategies + SA	Fuzzy + NN trained with Circuit simulator	✓	✗	✗	⃠/45 min
Shoou-Jin [104]	Passive filters	GA	Equations	✗	✓[5]	✗	⃠/⃠
Barros [21, 22]	Analog cells	GA	Circuit simulator and feasibility SVM models	✓	✗	✗	⃠/20 min
Castro-Lopez [19]	Amplifier	SA + Powel's method	Circuit simulator	⃠	✗	✓	⃠/25 min
Santos-Tavares [24]	Amplifier	Parallel GA	Circuit simulator	⃠	✗	✗	⃠/⃠
MOJITO [36]	Amplifier	NSGA-II	Circuit simulator	✓	✓[5]	✗	⃠/<7 days
Pradhan [37]	Amplifier, Filter	Multi-objective SA	Layout aware MNA models	✗	✗	✗	⃠/16 min
Matsukawa [49]	ADC	Convex Optimization	Convex functions	✓	✓[5]	✗	⃠/⃠
Cheng [17]	Amplifier	SA	Equations	✗	✗	✓	⃠/<1 h
Hongying [29]	Amplifier	GA with VDE	Circuit simulator	✗	✓[5]	✗	⃠/⃠
Fakhfakh [35]	LNA current conveyor	Multi-Objective PSO	Equations	✗	✗	✗	⃠/⃠

(continued)

Table 2.3 (continued)

Tool\author	Circuits	Design plan/optim.	Evaluation	1	2	3	Time setup/exec.
Kuo-Hsuan [14]	RFDA	Convex optimization Stochastic Fine Tuning	Posynomial models Circuit simulator	✘	✘	✘	⃠/1 h
Habal [84]	Tunable OTA	Deterministic non-linear optimizer	Circuit simulator	✘	✘	✓	⃠/2–11 h
Kamisetty [33]	Oscillator	PSO	Equations	✘	✘	✘	⃠/ ⃠
Benhala [32]	Current Conveyors	ACO	Equations	✘	✘	✘	⃠/<1 min
Roca [40]	Amplifiers	NSGA-II	Circuit simulator	✘	✘	✘	⃠/1–2 h
Gupta and Gosh [31]	Amplifiers	ACO	Circuit simulator	✘	✘	✘	⃠/1–3 h
Kumar and Duraiswamy [34]	Amplifier	PSO	Circuit simulator	✘	✘	✘	⃠/ ⃠
AIDA-C [62, 93, 94]	OPAMPs	NSGA-II	Circuit simulator and SVM	✓	✘	✘	1 h/several hours
AIDA-GM [96–98]	LNA OPAMPs	NSGA-II with gradient models	Circuit simulator	✓	✘	✘	1 h/several hours
Liao et al. [81]	OPAMP	Nonlinear programming	Bias driven equations	✓	✘	✓	⃠/<1 s
Afacan et al. [57]	Active Filter	Evolutionary strategies	Equations and Simulator	✘	✘	✘	⃠/<few minutes
Chen et al. [69]	OPAMPS	Convex Solver	Bias driven equations with WCD	✓	✓	✘	⃠/<1 s
AIDA [87, 95, 98]	Oscillators OPAMPs	NSGA-II; MOPSO; MOSA	Circuit simulator	✓	✘	✓	1 h/several days

✓—Supported; ✘—Not supported; ⃠—Undefiend

1 Robust design; *2* Topology; *3* Includes layout effects

[a]Topology is handled before sizing; [b]Topology is handled during sizing; [c]Layout is handled after sizing

Circuit Evaluation In the tool that implements the proposed techniques, AIDA, the evaluation of the circuits is done using the circuit simulator. In equation-based systems the usage of classical optimization methods is possible, however, the accuracy of the models and the derivation of such equations strong limits its applicability. Besides the accuracy ensured in the evaluation, the generality of the approach is increased and the setup time required to introduce new circuits is decreased, as the circuit simulator is part of the traditional design flow anyway. The price for these advantages is paid with a significantly increase in the evaluation time, and also, the relation between the performance figures and the design variables becomes unknown, making the usage of classic optimization methods inappropriate.

Optimization Algorithms AIDA uses a multi-objective multi-constraint optimization based on evolutionary techniques. From the surveyed simulation-based approaches the most common stochastic algorithms were based on simulation annealing and evolutionary approaches, with some of the latest implementations considering swarm intelligence methods (PSO and ACO). Besides, analog IC design is characterized by highly constrained problems and many tradeoffs. In this context, special relevance is given to multi-objective algorithms.

2.5.1 Contributions

The contributions that were described in Chap. 1 of this book are here revisited in light of the techniques used in state-of-the-art in analog IC sizing automation.

First, in [93], a simple but effective method to model the analog IC sizing optimization in multi-objective multi-constraint optimization using NSGA-II is proposed. The model accounts for variability by considering worst case corners. In the implemented tool, the worst case corners are selected by the designer. The main advantage of this approach to handle variability is its simplicity. Besides, the usage of corner sets is a common signoff practice in the industry. However, by delegating to the designer the selection of the relevant corners, a place for human error is added to the design flow. To automatically derive the worst case corners advanced techniques like Fast PVT, sigma driven corners or WCD could be introduced in the implementation. The relevance of true multi-objective analog IC sizing and optimization is further explored in [94] where a VCO is optimized for both minimum Phase-Noise and Power consumption, instead of the traditional optimization of the figure-of-merit (FOM) that relate both objectives. In this work the obtained POF contains more than 40 different solutions with performance that beat current state-of-the-art oscillators and delivers the optimal tradeoffs to the designer.

As seen, the main drawback of using the circuit simulator is the increased evaluation time, hence an effective usage of the simulations is mandatory. To generate better tentative solutions, in [95–97] the Gradient Model, which is automatically extracted using machine learning techniques, is used to add circuit specific knowledge to the optimization. The Gradient Model, is embedded in the

implementation of the genetic operators of the optimization kernel, increasing the efficiency of the exploration of the search space.

While many algorithms have been used to approach analog IC sizing, a fair comparison between approaches was not presented in the state-of-the-art. Historically, both SA and GA have been used intensively, in this sense is natural to consider both evolutionary and annealing approaches. Given the recent experiments with swarm intelligence in this domain, and to broad the scope of the implemented framework, this class of methods were also considered. From a brief perusal of the multi-objectives implementations, the ideas like non-sorted domination or crowding distance (further described in Chap. 4 of this book) presented in NSGA-II are reused by several other methods. In terms of the swarm intelligence algorithms, both ACO and PSO have been applied to circuit sizing. Because of MOPSO is already found in the literature and the un-natural application to real valued problems of the path finding ideas of the ACO, MOPSO was considered. In [98] these stochastic approaches were experimented in for the sizing of oscillators.

In [87–89] and in Chap. 6 of this book is presented a methodology for automatic layout-aware sizing of analog ICs that uses a built-in extractor to accurately compute the impact of parasitic from both floorplan and early-stages of routing. The computational efficiency is improved by avoiding the time-consuming detailed layout generation needed by off-the-shelf tools for parasitic extraction [99]. Moreover, prior knowledge of the circuit's parasitics is not required, i.e., there is no need for circuit specific equations or models as in [79]. The applicability of the approach is demonstrated with post-layout performances attained for the circuits designed with the developed layout-aware methodology, versus, the traditional optimization-based sizing.

2.6 Conclusions

Although much has been accomplished in automatic design of analog circuits, the fact is that automatic custom generators usable in industrial design environment are just starting to gain ground. As seen, the evolution of the technology keeps creating new challenges/opportunities for analog EDA researchers. In this survey, the most significant analog IC sizing automation tools were presented and analyzed to provide a better understanding of its advantages and shortcomings. The different approaches were classified in terms of the techniques used and the most significant aspects observed were the setup and the execution time, as well as the accuracy in the evaluation of the solutions. The trends in optimization methods were also surveyed, showing a predominance of the multi-objective approaches.

The presented study was used to help and frame the definition of the techniques and strategies used in the development of AIDA, which is described in the next chapters.

References

1. Barros MFM, Guilherme JMC, Horta NCG (2010) Analog circuits and systems optimization based on evolutionary computation techniques. Springer, Berlin
2. Degrauwe MGR et al (1987) IDAC: an interactive design tool for analog CMOS circuits. IEEE J Solid-State Circuits 22(6):1106–1116
3. Harjani R, Rutenbar RA, Carley LR (1989) OASYS: a framework for analog circuit synthesis. IEEE Trans Comput Aided Des Integr Circuits Syst 8(12):1247–1266
4. El-Turky F, Perry EE (1989) BLADES: an artificial intelligence approach to analog circuit design. IEEE Trans Comput Aided Des Integr Circuits Syst 8(6):680–692
5. Sheu BJ, Lee JC (1990) Flexible architecture approach to knowledge-based analogue IC design. IEE Proc G—Circuits Dev Syst 137(4):266–274
6. Makris CA, Toumazou C (1995) Analog IC design automation. II. Automated circuit correction by qualitative reasoning. IEEE Trans Computer-Aided Design Integr Circuits Syst 14(2):239–254
7. Toumazou C, Makris CA (1995) Analog IC design automation. I. Automated circuit generation: new concepts and methods. IEEE Trans Comput Aided Des Integr Circuits Syst 14(2):218–238
8. Nagel LW (1975) SPICE2: a computer program to simulate semiconductor circuits. Thesis, EECS Department, University of California, Berkeley
9. Gielen GGE, Rutenbar RA (2000) Computer-aided design of analog and mixed-signal integrated circuits. Proc IEEE 88(12):1825–1852
10. Cadence (2014) Cadence. http://www.cadence.com/. Accessed 17 Oct 2014
11. Synopsys (2014) Synopsys. http://www.synopsys.com/. Accessed 17 Oct 2014
12. MunEda (2015) MunEDA. http://www.muneda.com/. Accessed 12 Apr 2015
13. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
14. Kuo-Hsuan M, Po-Cheng P, Hung-Ming C (2011) Integrated hierarchical synthesis of analog/RF circuits with accurate performance mapping. In: International symposium on quality electronic design, Santa Clara, 2011
15. Torralba A, Chavez J, Franquelo LG (1996) FASY: a fuzzy-logic based tool for analog synthesis. IEEE Trans Comput Aided Des Integr Circuits Syst 15(7):705–715
16. Torralba AJ, Chavez J, Franquelo LG (1996) Fuzzy-logic-based analog design tools. IEEE Micro 16(4):60–68
17. Cheng-Wu L, Pin-Dai S, Ya-Ting S, Soon-Jyh C (2009) A bias-driven approach for automated design of operational amplifiers. In: International symposium on VLSI design, automation and test, Hsinchu, 2009
18. Medeiro F, Fernandez FV, Dominguez-Castro R, Rodriguez-Vazquez A (1994) A statistical optimization-based approach for automated sizing of analog cells. In IEEE/ACM international conference on computer-aided design (ICCAD), 1994
19. Castro-Lopez R, Guerra O, Roca E, Fernandez FV (2008) An integrated layout-synthesis approach for analog ICs. IEEE Trans Comput Aided Des Integr Circuits Syst 27 (7):1179–1189
20. Goldberg David E (1994) Genetic and evolutionary algorithms come of age. Commun ACM 37(3):113–119
21. Barros M, Guilherme J, Horta N (2010) Analog circuits optimization based on evolutionary computation techniques. Integr VLSI J 43(1):136–155
22. Barros M, Guilherme J, Horta N (2007) GA-SVM feasibility model and optimization kernel applied to analog IC design automation. In: ACM Great Lakes symposium on VLSI, Stresa-Lago Maggiore, 2007
23. Alpaydin G, Balkir S, Dundar G (2003) An evolutionary approach to automatic synthesis of high-performance analog integrated circuits. IEEE Trans Evol Comput 7(3):240–252

24. Santos-Tavares R, Paulino N, Higino J, Goes J, Oliveira JP (2008) Optimization of multi-stage amplifiers in deep-submicron CMOS using a distributed/parallel genetic algorithm. In: International symposium on circuits and systems, Seattle, 2008
25. Krasnicki M, Phelps R, Rutenbar RA, Carley LR (1999) MAELSTROM: efficient simulation-based synthesis for custom analog cells. In: Design automation conference, New Orleans, 1999
26. Phelps R, Krasnicki M, Rutenbar RA, Carley LR, Hellums JR (2000) Anaconda: simulation-based synthesis of analog circuits via stochastic pattern search. IEEE Trans Comput Aided Des Integr Circuits Syst 19(6):703–717
27. Koza JR, III Bennett FH, Andre D, Keane MA, Dunlap F (1997) Automated synthesis of analog electrical circuits by means of genetic programming. IEEE Trans Evol Comput 1 (2):109–128
28. Sripramong T, Toumazou C (2002) The invention of CMOS amplifiers using genetic programming and current-flow analysis. IEEE Trans Comput Aided Des Integr Circuits Syst 21(11):1237–1252
29. Hongying Y, Jingsong H (2010) Evolutionary design of operational amplifier using variable-length differential evolution algorithm. In: International conference on computer application and system modeling, Taiyuan Shanxi, 2010
30. Chu S-C, Huang H-C, Roddick JF, Pan J-S (2011) Overview of algorithms for swarm intelligence. In: Jedrzejowicz P, Nguyen NT, Hoang K (eds) Computational collective intelligence. Technologies and applications, Third International Conference, ICCCI 2011, Gdynia, Poland, September 21–23, 2011, Proceedings, Part I. Lecture Notes in Computer Science, vol 6922. Springer, Heidelberg, pp 28–41
31. Gupta H, Ghosh B (2012) Analog circuits design using ant colony optimization. Int J Electron Comput Commun Technol 2(3):9–21
32. Benhala B, Ahaitouf A, Fakhfakh M, Mechaqrane A (2012) New adaptation of the ACO algorithm for the analog circuits design optimization. Int J Comput Sci Issues 9(3):360–367
33. Kamisetty S, Garg J, Tripathi JN, Mukherjee J (2011) Optimization of analog RF circuit parameters using randomness in particle swarm optimization. In: World Congress on information and communication technologies, Mumbai, 11–14 Dec 2011
34. Kumar PP, Duraiswamy K (2012) An optimized device sizing of analog circuits using particle swarm optimization. J Comput Sci 8(6):930–935
35. Fakhfakh Mourad, Cooren Yann, Sallem Amin, Loulou Mourad, Siarry Patrick (2010) Analog circuit design optimization through the particle swarm optimization technique. Analog Integr Circ Sig Process 63(1):71–82
36. McConaghy T, Palmers P, Steyaert M, Gielen G (2011) Trustworthy genetic programming-based synthesis of analog circuit topologies using hierarchical domain-specific building blocks. IEEE Trans Evol Comput 15(4):557–570
37. Pradhan A, Vemuri R (2009) Efficient synthesis of a uniformly spread layout aware pareto surface for analog circuits. In: International conference on VLSI Design, New Delhi, 2009
38. Deniz E, Dundar G (2010) Hierarchical performance estimation of analog blocks using Pareto Fronts. In: 2010 Conference on Ph.D. research in microelectronics and electronics (PRIME), Berlin, 18–21 July 2010
39. Castro-Lopez R, Roca E, Fernandez FV (2009) Multimode Pareto fronts for design of reconfigurable analogue circuits. Electron Lett 45(2):95–96
40. Roca Elisenda, Velasco-Jiménez Manuel, Castro-López Rafael, Fernández Francisco V (2012) Context-dependent transformation of pareto-optimal performance fronts of operational amplifiers. Analog Integr Circ Sig Process 73(1):65–76
41. Gielen G, McConaghy T, Eeckelaert T (2005) Performance space modeling for hierarchical synthesis of analog integrated circuits. In: 42nd design automation conference, 13–17 June 2005
42. Koh HY, Sequin CH, Gray PR (1990) OPASYN: a compiler for CMOS operational amplifiers. IEEE Trans Comput Aided Des Integr Circuits Syst 9(2):113–125

43. Harvey JP, Elmasry MI, Leung B (1992) STAIC: an interactive framework for synthesizing CMOS and BiCMOS analog circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 11 (11):1402–1417
44. Gielen GGE, Walscharts HCC, Sansen WMC (1990) Analog circuit design optimization based on symbolic simulation and simulated annealing. IEEE J Solid-State Circuits 25 (3):707–713
45. Gielen GGE, Walscharts HCC, Sansen WMC (1990) ISAAC: a symbolic simulator for analog integrated circuits. IEEE J Solid-State Circuits 25(3):707–713
46. Swings K, Sansen W (1991) DONALD: a workbench for interactive design space exploration and sizing of analog circuits. In: Proceedings of the European conference on design automation (EDAC), Amsterdam, 25–28 Feb 1991
47. Maulik PC, Carley LR, Allstot DJ (1993) Sizing of cell-level analog circuits using constrained optimization techniques. IEEE J Solid-State Circuits 28(3):223–241
48. Maulik PC, Carley LR, Rutenbar RA (1995) Integer programming based topology selection of cell-level analog circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 14 (4):401–412
49. Matsukawa K et al (2009) Design methods for pipeline and delta-sigma A-to-D converters with convex optimization. In: 2009 Asia and South Pacific design automation conference, Yokohama, 19–22 Jan 2009
50. Ochotta ES, Rutenbar RA, Carley LR (1996) Synthesis of high-performance analog circuits in ASTRX/OBLX. IEEE Trans Comput Aided Des Integr Circuits Syst 15(3):273–294
51. Kruiskamp W, Leenaerts D (1995) DARWIN: CMOS opamp synthesis by means of a genetic algorithm. In: Proceedings of the 32nd annual ACM/IEEE design automation conference DAC'95, San Francisco, CA, 1995
52. del Mar Hershenson M, Boyd SP, Lee TH (1998) GPCAD: a tool for CMOS op-amp synthesis. In: IEEE/ACM international conference on computer-aided design, San Jose, 1998
53. Gielen G, Wambacq P, Sansen WM (1994) Symbolic analysis methods and applications for analog circuits: a tutorial overview. Proc IEEE 82(2):680–692
54. Daems W, Gielen G, Sansen W (2003) Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 22(5):517–534
55. Nye W, Riley DC, Sangiovanni-Vincentelli A, Tits AL (1988) DELIGHT.SPICE: an optimization-based system for the design of integrated circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 7(4):501–519
56. Massier T, Graeb H, Schlichtmann U (2008) The sizing rules method for CMOS and bipolar analog integrated circuit synthesis. IEEE Trans Comput Aided Des Integr Circuits Syst 27 (12):2209–2222
57. Afacan E, Ay S, Fernandez FV, Dundar G, Basckaya F (2014) Model based hierarchical optimization strategies for analog design automation. In: 2014 Design, automation and test in Europe conference and exhibition (DATE), Dresden, 24–28 March 2014
58. Yelten MB, Ting Zhu, Koziel S, Franzon PD, Steer MB (2012) Demystifying surrogate modeling for circuits and systems. IEEE Circuits Syst Mag 12(1):45–63
59. Liu B, Zhang Q, Gielen G (2013) A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems. IEEE Trans Evol Comput 18 (9):180–192
60. De Bernardinis F, Jordan MI, Sangiovanni Vincentelli A (2004) Support vector machines for analog circuit performance representation. In: Proceedings of the 40th annual Design Automation Conference (DAC'03), Anaheim CA, 2–6 June 2003
61. Wolfe GA (2004) Performance macro-modeling techniques for fast analog circuit synthesis. Dissertation, University of Cincinnati
62. Lourenço N, Martins R, Barros M, Horta N (2013) Analog circuit design based on robust POFs using an enhanced MOEA with SVM models. In: Fakhfakh M, Tlelo-Cuautle E, Castro-Lopez R (eds) Analog/RF and mixed-signal circuit systematic design. Springer. Lecture Notes in Electrical Engineering, vol 233. Spring, Heidelberg, pp 149–167

63. McConaghy T, Breen K, Dyck J, Gupta A (2013) Variation-aware design of custom integrated circuits: a hands-on field guide. Springer-Verlag, New York
64. Debyser G, Gielen G (1998) Efficient analog circuit synthesis with simultaneous yield and robustness optimization. In: 1998 IEEE/ACM international conference on computer-aided design (ICCAD 98), San Jose, CA, 8–12 Nov 1998
65. Dharchoudhury A, Kang SM (1995) Worst-case analysis and optimization of VLSI circuit performances. IEEE Trans Comput Aided Des Integr Circuits Syst 14(4):481–492
66. Antreich KJ, Graeb HE, Wieser CU (1994) Circuit analysis and optimization driven by worst-case distances. IEEE Trans Comput Aided Des Integr Circuits Syst 13(1):57–71
67. Schwencker R, Schenkel F, Pronath M, Graeb H (2002) Analog circuit sizing using adaptive worst-case parameter sets. In: Design, automation and test in Europe conference and exhibition, Paris, 4–8 March 2002
68. Graeb HE (2007) Analog design centering and sizing. Springer, Netherlands
69. Chen YL, Wu WR, Liu CNJ, Li JCM (2014) Simultaneous optimization of analog circuits with reliability and variability for applications on flexible electronics. IEEE Trans Comput Aided Des Integr Circuits Syst 33(1):24–35
70. McConaghy T, Gielen GGE (2009) Globally reliable variation-aware sizing of analog integrated circuits via response surfaces and structural homotopy. IEEE Trans Comput Aided Des Integr Circuits Syst 28(11):1627–1640
71. Liu Bo, Fernandez FV, Bo Liu FV, Gielen GGE (2011) Efficient and accurate statistical analog yield optimization and variation-aware circuit sizing based on computational intelligence techniques. IEEE Trans Comput Aided Des Integr Circuits Syst 30(6):793–805
72. Singhee A, Rutenbar RA (2009) Statistical blockade: very fast statistical simulation and modeling of rare circuit events and its application to memory design. IEEE Trans Comput Aided Des Integr Circuits Syst 28(8):1176–1189
73. Kanj R, Joshi R, Nassif S (2006) Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare failure events. In: 43rd ACM/IEEE design automation conference, San Francisco, CA, 2006
74. Sun S, Li X, Liu H, Luo K, Gu B (2015) Fast statistical analysis of rare circuit failure events via scaled-sigma sampling for high-dimensional variation space. IEEE Trans Comput Aided Des Integr Circuits Syst 34(7):1096–1109
75. Maricau E, Gielen G (2011) Computer-aided analog circuit design for reliability in nanometer CMOS. IEEE J Emerg Selected Topics Circuits Syst 1(1):50–58
76. Pan X, Graeb H (2010) Reliability analysis of analog circuits using quadratic lifetime worst-case distance prediction. In: 2010 IEEE custom integrated circuits conference (CICC), San Jose CA, 19–22 Sept 2010
77. Solido Design Automation (2015) Solido design automation http://www.solidodesign.com/. Accessed 12 April 2015
78. Hastings A (2005) The art of analog layout, 2nd edn. Prentice Hall, Upper Saddle River
79. Graeb HE (ed) (2010) Analog layout synthesis. Springer, Berlin
80. Ranjan M et al (2004) Fast, layout-inclusive analog circuit synthesis using pre-compiled parasitic-aware symbolic performance models. In: Design, automation and test in europe conference and exhibition, Paris, 2004
81. Liao YC, Chen YL, Cai XT, Liu CN, Chen TC (2013) LASER: layout-aware analog synthesis environment on laker. In: Proceedings of the 23rd ACM international conference on Great lakes symposium on VLSI, Paris, 2–4 May 2013
82. Vancorenland P, Van der Plas G, Steyaert M, Gielen G, Sansen W (2001) A layout-aware synthesis methodology for RF circuits. In: IEEE/ACM international conference computer-aided design, 2001
83. Youssef S, Javid F, Dupuis D, Iskander R, Louerat M-M (2011) A python-based layout-aware analog design methodology for nanometric technologies. In: 2011 IEEE 6th international design and test workshop (IDT), Beirut, 11–14 Dec 2011
84. Habal H, Graeb H (2011) Constraint-based layout-driven sizing of analog circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 30(8):1089–1102

85. Toro-Frias A, Castro-Lopez R, Fernandez F (2012) An automated layout-aware design flow. In: International conference on synthesis, modeling, analysis and simulation methods and applications to circuit design (SMACD), Seville, 19–21 September 2012
86. Berkol G, Unutulmaz A, Afacan E, Dundar G, Fernandez FV, Pusane AE, Baskaya F (2015) A Two-step layout-in-the-loop design automation tool. In IEEE 13th international new circuits and systems conference (NEWCAS), Grenoble, 7–10 June 2015
87. Lourenço N, Canelas A, Póvoa R, Martins R, Horta N (2014) Floorplan-aware analog IC sizing and optimization based on topological constraints. Integration, the VLSI Journal, Elsevier 48:183–197
88. Lourenco N, Martins R, Horta N (2015) Layout-aware sizing of analog ICs using floorplan and routing estimates for parasitic extraction. In: 2015 Design, automation and test in europe conference and exhibition (DATE), Grenoble, 9–13 March 2015
89. Martins R, Lourenco N, Canelas A, Povoa R, Horta N (2015) AIDA: Robust layout-aware synthesis of analog ICs including sizing and layout generation. In: International conference on synthesis, modeling, analysis and simulation methods and applications to circuit design (SMACD), Istanbul, 7–9 Sept 2015
90. Paydavosi N et al BSIM4v4.8.0 MOSFET Model Manual. http://www-device.eecs.berkeley.edu/bsim/Files/BSIM4/BSIM480/BSIM480_Manual.pdf. Accessed 12 April 2016
91. Strasser M, Eick M, Grab H, Schlichtmann U, Johannes FM (2008) Deterministic analog circuit placement using hierarchically bounded enumeration and enhanced shape functions. In IEEE/ACM international conference computer-aided design, San Jose, 2008
92. Doboli A, Dhanwada N, Nunez-Aldana A, Vemuri R (2004) A two-layer library-based approach to synthesis of analog systems from VHDL-AMS specifications. ACM Trans Des Autom Electron Syst 9(2):238–271
93. Lourenço N, Horta N (2012) GENOM-POF: multi-objective evolutionary synthesis of analog ICs with corners validation. In: Genetic and evolutionary computation conference, Philadelphia, 2012
94. Póvoa R, Lourenço R, Lourenço N, Canelas A, Martins R, Horta N (2014) LC-VCO automatic synthesis using multi-objective evolutionary techniques. In: 2014 IEEE international symposium on circuits and systems (ISCAS), Melbourne VIC, 2014
95. Rocha F, Martins R, Lourenço N, Horta N (2014) Electronic design automation of analog ics combining gradient models with multi-objective evolutionary algorithms. Springer International Publishing
96. Rocha F, Lourenço N, Póvoa R, Martins R, Horta N (2014) A new metaheuristic combining gradient models with NSGA-II to enhance analog IC synthesis. In: 2013 IEEE congress on evolutionary computation, Cancun, 2013
97. Rocha F, Martins R, Lourenço N, Horta N (2013) Enhancing a layout-aware synthesis methodology for analog ICs by embedding statistical knowledge into the evolutionary optimization Kernel. In: Camarinha-Matos L, Tomic S, Graça P (eds) Technological innovation for the internet of things. 4th IFIP WG 5.5/SOCOLNET Doctoral Conference on computing, electrical and industrial systems, DoCEIS 2013, Costa de Caparica, Portugal, April 15–17, 2013. Proceedings, IFIP advances in information and communication technology, vol 394. Springer Berlin Heidelberg
98. Póvoa R, Lourenço R, Lourenço N, Canelas A, Martins R, Horta N (2015) Synthesis of LC-oscillators using rival multi-objective/multi-constraint optimization kernels. In: Fakhfakh M, Tlelo-Cuautle E, Fino M (eds) Performance optimization techniques in analog, mixed-signal, and radio-frequency circuit design. IGI Global, Hershey PA, pp 1–27
99. Martins R, Lourenço N, Canelas A, Horta N (2014) Electromigration-aware analog router with multilayer multiport terminal structures. Integration, VLSI J 47(4):532–547
100. Ning ZQ, Mouthaan T, Wallinga H (1991) SEAS: a simulated evolution approach for analog circuit synthesis. In: Custom integrated circuits conference, 1991
101. Chang H et al (1992) A top-down, constraint-driven design methodology for analog integrated circuits. In: Custom integrated circuits conference, 1992

102. Maulik PC, Carley LR, Rutenbar RA (1995) Integer programming based topology selection of cell-level analog circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 14 (4):401–412
103. Medeiro F, Perez-Verdu B, Rodriguez-Vazquez A, Huertas JL (1995) A vertically integrated tool for automated design of ΣΔ modulators. IEEE J Solid-State Circuits 30(7):762–772
104. Chang SJ, Hou HS, Su YK (2006) Automated passive filter synthesis using a novel tree representation and genetic programming. IEEE Trans Evol Comput 10(1):93–100

Chapter 3
AIDA-C Architecture

3.1 AIDA Environment

Analog IC design automation (AIDA) framework [1–3] results from the integration of the analog IC design automation tools, AIDA-C and AIDA-L, as illustrated in Fig. 3.1. AIDA-C, the focus of this book, targets sizing of the devices in analog circuits using state-of-the-art and innovative techniques. Its optimizer kernel is based in multi-objective multi-constraint optimization techniques and, the tool addresses robust design requirements in a worst case approach, by considering user defined worst case corners. For accurate circuit's performance evaluation, industrial grade circuit simulators, such as Cadence® Spectre®, Mentor Graphics' ELDO™ or Synopsys® HSPICE®, are considered.

After the circuit-level design, the layout generator AIDA-L [4], takes as inputs the device sizes and corresponding floorplan templates, and generates the layout by placing and routing the devices using internal design-rule check (DRC) and layout-versus-schematic (LVS) procedures, completing the design flow. For simple cells, parametric generators are a valid solution to implement the layout generator, as found in the literature [5], however, parametric generators are usually specific to a technology, support only a handful of design options and their performance degrades fast for wide variations in the sizes of the devices, making them difficult to reuse. As such, for cells of moderate complexity, the development of effective parametric generators has proven ineffective either on design-time or on design-reusability.

In order to cope with these requirements and increase design efficiency, AIDA-L stores the floorplan regularities, i.e., impositions of the designer, in a layout meta-description that is independent of technology and sizing. The generated circuit layout follows those guidelines, and is saved as a GDSII stream format. A final physical verification of the results is performed using an external tool, e.g.,

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0_3

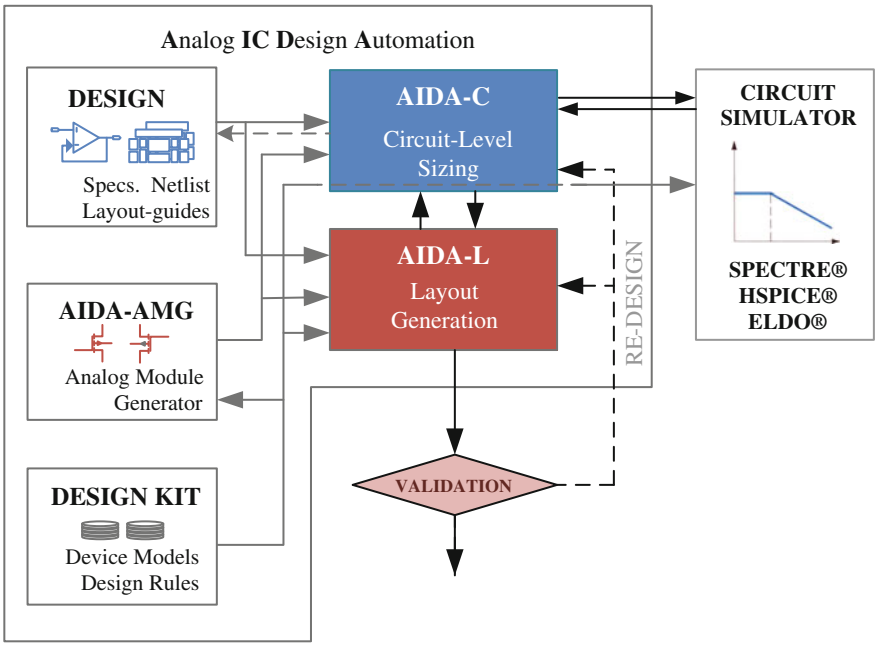


Fig. 3.1 AIDA overview

Mentor Graphics' Calibre® [6], to ensure industrial level design-rule check and layout-versus-schematic, and also, post-layout performance compliance.

Another important aspect is that AIDA-C interacts with AIDA-L's Placer to perform a floorplan-aware circuit sizing optimization, enabling the inclusion of geometrical information from the layout during the sizing optimization. AIDA-L's Placer is able to quickly generate the precise floorplan for any possible sizing solution in-the-loop, without significantly adding computation time. In addition, for an accurate estimation of the layout's parasitic devices, which are included in the original circuit's netlist enabling layout-aware circuit sizing, the AIDA-L's fully automatic router is used. In this process, AIDA-C's simulation results are used to feed the current/temperature pairs for the devices' terminals, which are needed for AIDA-L's electromigration-aware wire planning.

The remaining of this chapter is devoted to AIDA-C's architecture and design flow, with special care on the detailed description of all the interaction with the designer. The tool's setup and its interactions with designers is commonly overlooked by analog automation methodologies, but is one of the main barriers for practical usage of automation solutions. The optimization algorithms used to efficiently explore the design will be described in Chaps. 4, and 5 will focus on the layout-aware circuit sizing.

3.2 AIDA-C Architecture

AIDA-C, whose block diagram is shown in Fig. 3.2, is composed of two macro-blocks: the Setup and Monitoring and the Optimizer. The Setup and Monitoring block is devoted to assist the designer through the design process and ease the usage of the circuit optimizer. The Optimizer solves the circuit sizing using multi-objective optimization techniques, where the circuit's performance is obtained using industrial circuit simulators (for electrical measures) and AIDA-L (for layout related measures).

3.2.1 Setup and Monitoring

Part of the setup of a design in AIDA-C is made at file level. The inputs from the designer are the circuit and test-bench(es) in the form of spice-like netlist(s). The netlist(s) must have the optimization variables as parameters, and must include means to measure the circuit's performances considered in the design objectives and target specifications. Intended corner's parameter sets for worst case evaluation are also included. In addition, the designer defines ranges for the optimization variables, design constraints and optimization objectives. If a layout-aware circuit sizing optimization is intended, AIDA-L's layout guides must also be provided. The **Setup Assistant** helps the designer and speed ups the tool's setup by generating

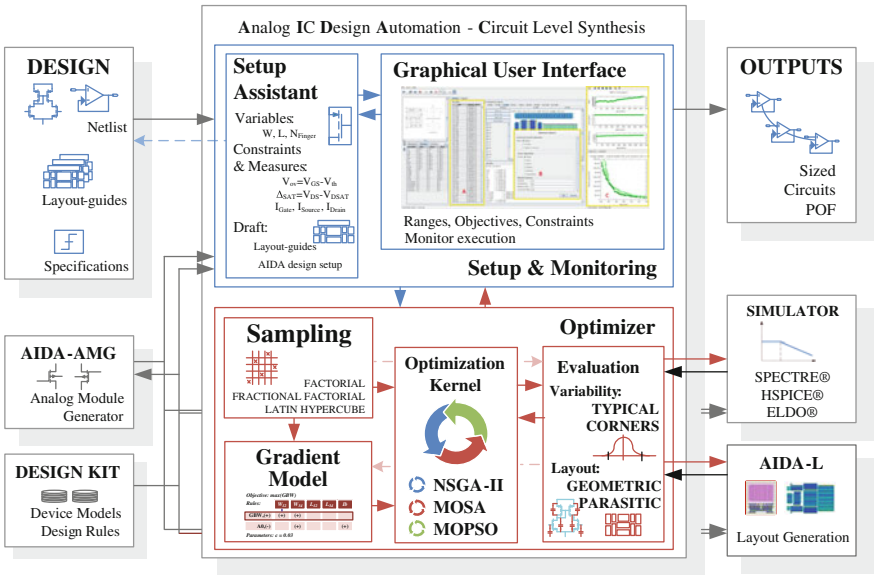


Fig. 3.2 AIDA-C architecture

some statements and file drafts automatically, for example, for the active devices some default measures and the correspondent constraint statements for overdrives and saturation margin (delta) are generated automatically. The measures for the currents in all circuit nodes required by the AIDA-L's wire-planner, are also generated by AIDA-C. In addition, a draft of the layout guides is generated from the netlist directly. Using these aids, the setup productivity is highly enhanced, without compromising the designer's ability to customize it.

In the **Graphical User Interface** (GUI) the designer manages previous designs (Loads/Saves previous solutions and configurations), sets the desired performance specifications and optimization targets, selects how the circuit is evaluated in terms of nominal or worst case simulation, and, what kind of layout effects are considered, if any. During the optimization, the designer monitors the evolution, and can intervene, e.g., by stopping the optimization whenever an acceptable solution is present, or, by editing a solution to make an educated "guess" that helps get to a better solution faster. Such interface permits an efficient reuse of the setup for different specifications, and, by using the circuit simulator as the core for performance evaluation, the integration in the traditional design flow is straightforward. In addition, as the interface was natively developed for the designer to interact with the optimizer, the optimizer efficiency can be further improved by the designer's own expertise.

3.2.2 Circuit Optimizer

AIDA-C's optimizer is centered on the multi-objective multi-constraint **optimization kernel**. The optimization methods implemented in AIDA-C are the NSGA-II [7], MOSA [8] and MOPSO [9]. Additionally, the algorithm implementations share a common interface, easing the intermingling of tentative solutions between technics in order to, not only use the different approaches by themselves, but also ease the combination of the methods.

Given the multi-objective nature of the sizing method, the optimizer's output is not one solution but a set of solutions all compliant with the design specifications, i.e., a set of Pareto non-dominated solutions or Pareto front. Pareto dominance states that one point in the solution space, A, is not dominated by another point, B, if $\exists m$: $f_m(\mathrm{A}) < f_m(\mathrm{B})$. Figure 3.3 depicts a Pareto front, illustrating the Pareto dominance concept, where solutions A and B are non-dominated and both dominate solution point C.

To ensure accuracy and generality of application, the **evaluation** of the electrical circuit performance is done using circuit simulators. AIDA-C supports two evaluation strategies, Typical and Corners. In Typical, the circuit is evaluated using only nominal conditions. In Corners, the design is optimized using all the defined worst case process, voltage and temperature (PVT) corners, i.e., for each evaluation

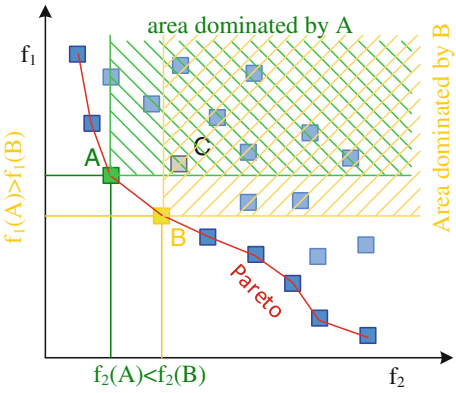


Fig. 3.3 Pareto front illustration

the circuit is simulated once for each corner case, and the worst case values are considered [4]. Geometric and performance figures that depend on the physical layout representation of the circuit are considered using AIDA-L to generate the layout and extract the corresponding parasitics.

AIDA-C is enhanced with the usage of machine learning techniques that add automatically generated design knowledge to the optimizer. The **Gradient Model** [10], which creates a simplistic model of the 1st order iterations between design variables and circuit performance, is used to guide the generation process to a better solution faster. The generation of the model starts by **sampling** the design space using a design of experiments (DOE) approach [11], and then, the model is generated from those samples by extracting and ranking the contributions of each design variable (input) to each design performance or objective (output). Finally, a set of gradient rules are built and combined with the evolutionary optimization kernel.

3.3 AIDA-C's Analog IC Design Flow

The detailed analog IC design flow based on the AIDA-C solution is illustrated in Fig. 3.4. The following paragraphs describe the flow, starting from the inputs until sizing is complete. The particulars of setting up a design for the implemented tool are provided for clear image of the amount of effort that is needed to setup such an automation tool, which can be quite small. But these steps are general enough to be used, with the proper adaptation, with other analog IC automation solutions. A single-ended two-stage Miller amplifier for United Microelectronics Corporation (UMC) 0.13 μm is used to illustrate the descriptions.

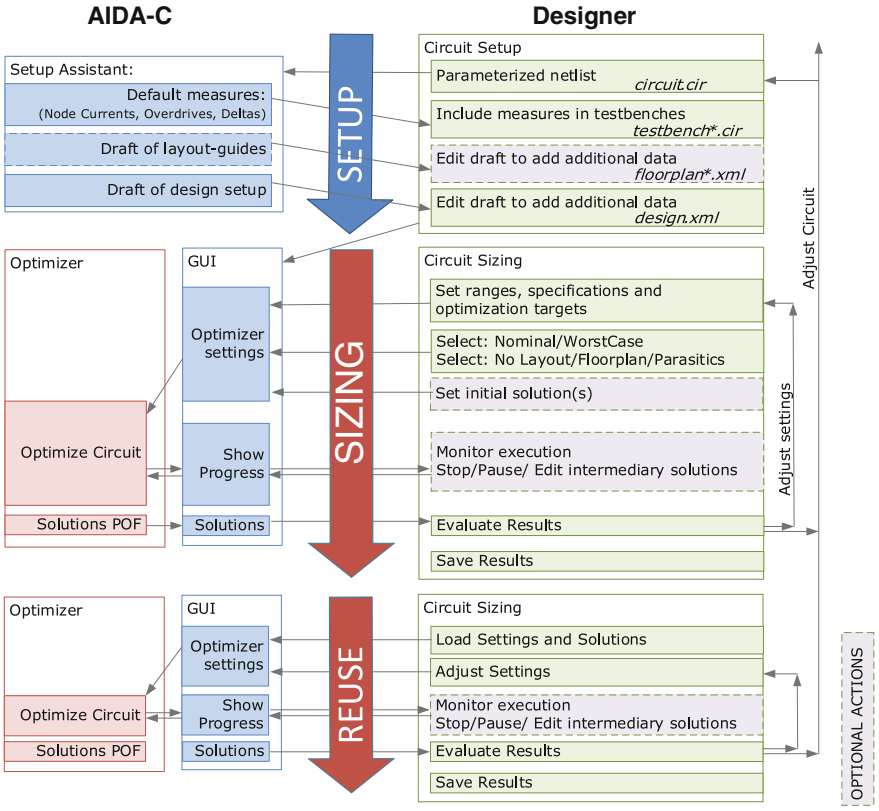


Fig. 3.4 Design flow using AIDA-C

3.3.1 Circuit Sizing Setup

In this section, it is explained how to setup all the necessary elements to use AIDA-C. Adding a circuit to AIDA-C is a two-step operation, first the circuit netlist needs to be properly parameterized, and then, using AIDA-C's setup assistant to accelerate the process, AIDA-C design structure, as shown in Fig. 3.5, needs to be created.

The project is organized by a set of files. The "design.xml" file is the main description file. XML is a markup language that defines a set of rules for encoding documents in a format that is both human- and machine-readable. The image is optional, and is used to show the circuit schematic to ease the identification of the devices and parameters when operating the tool (commonly a screenshot of the schematic). The folder "layout" contains multiple layout guide files and the folder "tech_netlist" contains the circuit and testbenches. The nominal and worst case (or

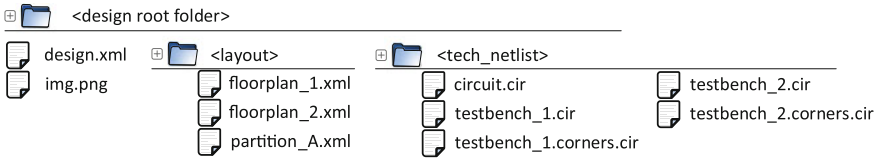


Fig. 3.5 AIDA-C design structure

corner) testbench(es) are described in different base files, but may include common files.

Starting from the circuit netlist, designed by hand or exported from a schematic editor, the design variables are defined by taking into consideration what are the values the optimizer may change (usually the sizes of the devices). At this stage, matching of devices can be enforced. The design variables are defined as parameters and are used in the netlist appropriately. Figure 3.6 shows the parameterized netlist used for the Miller amplifier example.

(a)

```
*** Beginning of Subcircuit definition
*** Cell name: Single-ended_two-stage_Miller_amplifier
.SUBCKT OPAMP_A DD IN IP OUT REF SS
  MP20 REF REF DD DD    P_12_HSL130E w=_wb l=_lb  m= 2     //w and l matched
  MP14 NETZ52 REF DD DD P_12_HSL130E w=_wb l=_lb  m=_mbp   //between MP20,
  MP22 OUT REF DD DD    P_12_HSL130E w=_wb l=_lb  m=_mb2   //MP14 and MP22
  MP11 D11 IN NETZ52 DD P_12_HSL130E w=_wp l=_lp  m=_mp //MP11 matched
  MP12 D12 IP NETZ52 DD P_12_HSL130E w=_wp l=_lp  m=_mp //with MP12
  MN9 D11 D11 SS SS    N_12_HSL130E  w=_wal l=_lal  m=_mal //MN9 matched
  MN10 D12 D11 SS SS   N_12_HSL130E  w=_wal l=_lal  m=_mal //with MN10
  MN21 OUT D12 SS SS   N_12_HSL130E  w=_w2g l=_l2g  m=_m2g
  XC0 OUT D12 SS momcaps_sy_mm l=_lc nf='_nfc'
.ENDS
*** End of Subcircuit definition
```

(b)

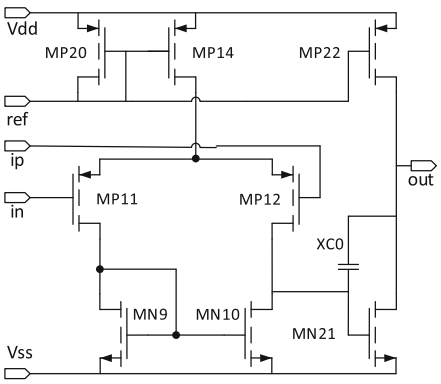


Fig. 3.6 Single-ended two-stage Miller amplifier. **a** Parameterized netlist (*circuit.cir*). **b** Schematic

```
<...>
************************** Analysis Selection ***************************
.OPTION AEX
.AC DEC 20 1 1G SWEEP DATA = PIPEdata
.LSTB V4
.NOISE V(OUTPUT, 0) V4 1

************************ Performance Measures ***************************
.EXTRACT DCAC label=IDD   -i(V0)
.EXTRACT AC   label=GDC   YVAL(LSTB_DB,1)
.MEASURE AC   label=GBW   when LSTB_DB=0 cross=1
.MEASURE AC   label=PHASE find LSTB_P when LSTB_DB=0
.EXTRACT AC   label=PM    PHASE + 180
.EXTRACT AC   label=PSRR  -YVAL(VDB(OUTPUT),1e6)
.EXTRACT DCAC label=SR2   -IDS(X1.mp22)/(CL)
.EXTRACT DCAC label=SR    MIN(SR2, SR2)
.EXTRACT DCAC label=VOFF  ABS(V(OUTPUT) - VCM)
.EXTRACT AC   label=No    RMS(ONOISE)
.EXTRACT AC   label=Sn    YVAL(INOISE,GBW)
.MEASURE AC   device_area = param('1e12*(_lc*_wc + _wb*_lb + _mbd*_wb*_lb +
      _mb2*_wb*_lb + 2*_mp*_wp*_lp + 2*_mal*_wal*_lal + _w2g*_l2g*_m2g)')
<...>
```

Fig. 3.7 AC testbench showing analysis and measures section (*testbench.cir*)

In addition to the circuit's netlist, testbenches with the statements to measure the circuit's performance must be added. AIDA-C parses the measures files obtained using the simulator's .measure/.extract commands to read the simulation results [12, 13]. Figure 3.7 shows the measures, options and analysis in the AC testbench used in the example. To increase performance, whenever possible multiple tentative sizing solutions are simulated in the same simulator call. In the illustrated testbench, the simulator SWEEP feature in the .AC analysis is used to achieve this. Alternatively, AIDA-C also supports .STEP and .ALTER [12, 13] to iterate over the several tentative solutions. For testbenches that use analyses that do not support loops, AIDA-C calls the simulator for each tentative solution. In addition to the native support for ELDO and HSPICE measure files, a default file-based interface that is used to interface with custom evaluation and parsers is available. AIDA-C's interface with SPECTRE was implemented using the referred interface.

The worst case simulation is done with alter statements, then, the worst value for each measure is considered when the optimizer evaluates the circuit's performance. The corner testbench is usually a copy of the typical case testbench with the additional *.alter* statements. Figure 3.8 shows the section of the AC testbench with definition of the corners.

From the parameterized netlist, the setup assistant automatically generates structural measures (overdrives, deltas and active area) for all transistors. In addition, DC current measures for all device terminals, which are needed for AIDA-L's electromigration-aware wiring planner, as referred, are also generated. These measures, shown in Fig. 3.9, are not mandatory, and is up to the designer to include them or not in the relevant testbenches.

At this point the netlist for the circuit and its testbenches are completed. The next step is to create the "design.xml" file where the circuit setup is described. Again, the

```
<...>
.ALTER wp_avdd_max_dvdd_max_temp_min_1_vcm_max
.lib '...' FF
.temp -55
.param VCM = vcm_max
.param VCC = '1.2*(1.05)'

.ALTER ws_avdd_min_dvdd_min_temp_max_8_vcm_max
.lib '...' SS
.TEMP 125
.param VCM = vcm_max
.param VCC = '1.2*(0.95)'

.ALTER wp_avdd_max_dvdd_max_temp_min_1_vcm_min
.lib '...' FF
.temp -55
.param VCM = vcm_min
.param VCC = '1.2*(1.05)'

.ALTER ws_avdd_min_dvdd_min_temp_max_8_vcm_min
.lib '...' SS
.TEMP 125
.param VCM = vcm_min
.param VCC = '1.2*(0.95)'
```

Fig. 3.8 AC testbench for corners showing.alter section (*testbench.corners.cir*)

```
***************** Transistor Bias Measures: OVERDRIVES ********************
.MEASURE DCAC vov_m20    = param('ABS(VGS(X1.mp20)-lv9(X1.mp20))')
.MEASURE DCAC vov_m14    = param('ABS(VGS(X1.mp14)-lv9(X1.mp14))')
.MEASURE DCAC vov_m22    = param('ABS(VGS(X1.mp22)-lv9(X1.mp22))')
<...>
***************** Transistor Bias Measures: MARGINS ************************
.MEASURE AC delta_m20    = param('ABS(VDS(X1.mp20)-VDSAT(X1.mp20))')
.MEASURE AC delta_m14    = param('ABS(VDS(X1.mp14)-VDSAT(X1.mp14))')
.MEASURE AC delta_m22    = param('ABS(VDS(X1.mp22)-VDSAT(X1.mp22))')
<...>
************************ Device Active Area ********************************
.MEASURE DCAC aa_mp20    = param('_wb*_lb*')
.MEASURE DCAC aa_mp14    = param('_wb*_lb*_mbp')
.MEASURE DCAC aa_mn22     = param('_wb*_lb*_mb2')
<...>
*************** IDC currents for EM-aware router *************************
.EXTRACT DCAC label=IDC_MP20_drain       I(X1.MP20.1)
.EXTRACT DCAC label=IDC_MP20_gate        I(X1.MP20.2)
.EXTRACT DCAC label=IDC_MP20_source      I(X1.MP20.3)
.EXTRACT DCAC label=IDC_MP20_bulk        I(X1.MP20.4)
.EXTRACT DCAC label=IDC_MP14_drain       I(X1.MP14.1)
.EXTRACT DCAC label=IDC_MP14_gate        I(X1.MP14.2)
.EXTRACT DCAC label=IDC_MP14_source      I(X1.MP14.3)
.EXTRACT DCAC label=IDC_MP14_bulk        I(X1.MP14.4)
<...>
```

Fig. 3.9 DC measures for ELDO™ AC testbench (*testbench*.cir*)

"setup assistant" is used to accelerate the setup by generating a draft of the "design.xml" from the data in the netlist and some technology-dependent default values (overdrives, ranges, etc.) that are stored in AIDA's design kit. The circuit netlist is parsed and the variables are detected, as well as devices. With this data, the setup assistant generates the partially filled design.xml shown in Fig. 3.10 for the 2-Stage

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Design SYSTEM "design-1.0.dtd">
<Design name=" OPAMP_A">
  <Circuit netlist="2stage.cir" techNode="UMC_013">
    <Layout template=""/>

    <Var id="_lc " range="4.4e-6:1e-7:1.0e-4"/>
    <Var id="_nfc " range="14:2:198"/>
    <Var id="_wb _wp _wal _w2g" range="1.00e-6:1e-7:10e-6"/>
    <Var id="_lb _lp _lal _l2g" range="0.12e-6:5e-8:10e-6"/>
    <Var id="_mbp _mb2 _mp _mal _m2g" range="1:2:1000"/>
  </Circuit>
  <TestbenchSetup simulator="eldo" inputMethod="LAM">
    <TestCase name=""/>

    <NominalTb netlist="">
      <Meas id="vov_mp20 vov_mp14 vov_mp22 vov_mp11 vov_mp12 vov_mp9
..." desc="Overdrive [V]"/>
      <Meas id="delta_mp20 delta_mp14 delta_mp22 delta_mp11 delta_mp12
..." desc="Delta [V]"/>
    </NominalTb>
    <WorstCaseTb netlist=".corners">
      <Meas importFrom=""/>
    </WorstCaseTb>
  </TestbenchSetup>
  <Constraint op="GE" value="0.10" meas="vov_mp20 vov_mp14 vov_mp22
vov_mp11 vov_mp12 vov_mn9 vov_mn10 vov_mn21" />
  <Constraint op="GE" value="0.10" meas=" delta_mp20 delta_mp14
delta_mp22 delta_mp11 delta_mp12 delta_mn9 delta_mn10 delta_mn21" />

</Design>
```

Fig. 3.10 Extract of the draft of circuit setup (*design.xml*)

Miller Amplifier. In addition, other fields like the corners and testbench declarations, which are required, are created with empty values to further speedup the setup.

Further automation is possible, but as the setup is not a common task, the current level of automation is enough for considerable savings in terms of setup, and also, helps avoiding some configuration errors. The designer is responsible for adding the missing data fields, and the complete setup for the Miller amplifier is shown in Fig. 3.11.

3.3.2 Layout-Aware Sizing Setup (Optional)

Optionally, when layout effects are to be considered, layout guides must be provided. The AIDA-L's layout guides, which are also described in XML format, describe the floorplan and are parameterized to include the design variables. The floorplan is defined using simple rectangular constructs that capture the proximity and topological relations that the designer wishes to impose. The information used for placement is the type and relative placement of the cells, and also, the symmetry, matching and combine requirements. The high level floorplan of each cell is described by a box shape, the size of this box has no meaning, only the relative position between cells (boxes) is of concern. The topological constraints that are enforced by the tool are inferred from the boxes' placement directly.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Design SYSTEM "design-1.0.dtd">
<Design name="OPAMP_A">
  <Circuit netlist="2stage.cir" techNode="UMC_013">
    <Layout template="t1a.xml"/>
  ...
    <Layout template="t3d.xml"/>

    <Var id="_lc" range="4.4e-6:1e-7:1.0e-4"/>
    <Var id="_nfc" range="14:2:198"/>
    <Var id="_wb _wp _wal _w2g" range="1.00e-6:1e-7:10e-6"/>
    <Var id="_lb _lp _lal _l2g" range="0.12e-6:5e-8:10e-6"/>
    <Var id="_mbp _mb2 _mp _mal _m2g" range="1:2:1000"/>
  </Circuit>
  <TestbenchSetup simulator="eldo" inputMethod="LAM">
    <TestCase name="wp_avdd_max_dvdd_max_temp_min_1_vcm_max"/>
    <TestCase name="ws_avdd_min_dvdd_min_temp_max_8_vcm_max"/>
    <TestCase name="wp_avdd_max_dvdd_max_temp_min_1_vcm_min"/>
    <TestCase name="ws_avdd_min_dvdd_min_temp_max_8_vcm_min"/>

    <NominalTb netlist=" ac_testbench.cir.eldo">
      <Meas name="idd" desc="I VDD [A]"/>
      <Meas name="gdc" desc="Gain DC [Hz]"/>
      <Meas name="gbw" desc="Unity Gain Frequency [Hz]"/>
      <Meas name="pm" desc="Phase margin [degrees]"/>
      <Meas name="psrr" desc="Power Supply Rejection Ratio [Hz]"/>
      <Meas name="sr" desc="Slew Rate [V/s]"/>
      <Meas name="voff" desc="Structural Offset [V]"/>
      <Meas name="no" desc="Noise [V]"/>
      <Meas name="sn" desc="Noise Density [V/sqrt(Hz)]"/>
      <Meas name="device_area" desc="Area from net list [m2]"/>
      <Meas id="vov_mp20 vov_mp14 vov_mp22 ..." desc="Overdrive [V]"/>
      <Meas id="delta_mp20 delta_mp14 delta_mp22 ..." desc="Delta [V]"/>
    </NominalTb>
    <WorstCaseTb netlist=" ac_testbench.cir.eldo.corners">
      <Meas importFrom="ac_testbench.cir.eldo"/>
    </WorstCaseTb>
  </TestbenchSetup>
  <Constraint op="GE" value="0.10" meas="vov_mp20 vov_mp14 vov_mp22
vov_mp11 vov_mp12 vov_mn9 vov_mn10 vov_mn21" />
  <Constraint op="GE" value="0.10" meas=" delta_mp20 delta_mp14
delta_mp22 delta_mp11 delta_mp12 delta_mn9 delta_mn10 delta_mn21" />
</Design>
```

Fig. 3.11 Completed of circuit setup (*design.xml*)

Symmetry is specified locally in each floorplan (or sub-floorplan) by two properties: 'symGroupId' that identifies a group of cells that share the same symmetry axe; and 'symCellId' that identifies a pair of the cells that are to be placed symmetrically in relation to the symmetry axe. By default, cells are self-symmetric and do not share their symmetry axe. Matching is enforced in the parameterization of the devices, where the devices that the designer deemed to be matched share some or all parameters (in both layout guides and netlist). Finally, combine operations can be used to replace a group of basic cells with more complex layout structures, e.g., merged structures or interdigitated/common-centroid layout styles.

In Fig. 3.12 the graphical representation of the hierarchically defined layout guides, and in Fig. 3.13 part of the corresponding XML description is shown. The devices in blue are defined hierarchically in the sub-template for partition P1A. The expected location of the power supply nets, VSS and VDD, are illustrated in the image running on top and bottom of the layout, respectively, although there is

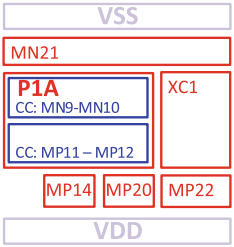


Fig. 3.12 Graphical representation of a template showing the relative location of the devices. (Reprinted from Integration, the VLSI Journal, 48, Nuno Lourenço, António Canelas, Ricardo Póvoa, Ricardo Martins, Nuno Horta, Floorplanaware analog IC sizing and optimization based on topological constraints, 183–197, Copyright (2015), with permission from Elsevier)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Template SYSTEM "template4.dtd">
<Template name="T1a">
 <CellList>
  <Cell name="P1" symGroupId="1">
   <Box x="-1250" y="1000" w="2500" h="1500" />
   <TemplateCellView file="p1a.xml" />
<!-- including part of pla.xml's CellList element inline-->
    <Cell name="MN9" symGroupId="2" symCellId="3">
     <Box x="-750" y="2000" w="1000" h="500" />
     <MOSFET type="N" width="wal*nfal" length="lal" nf="nfal" nrows="nral" />
    </Cell>
    <Cell name="MN10" symGroupId="2" symCellId="3">
     <Box x="750" y="2000" w="1000" h="500" />
     <MOSFET type="N" width="wal*nfal" length="lal" nf="nfal" nrows="nral" />
    </Cell>
    <Combine id0="MN9" id1="MN10" />
    <Cell name="MP11" symGroupId="2" symCellId="1">
      <Box x="-750" y="1000" w="1000" h="500" />
          <MOSFET type="P" width="wp*nfp" length="lp" nf="nfp" nrows="nrp" />
       </Cell>
       <Cell name="MP12" symGroupId="2" symCellId="1">
         <Box x="750" y="1000" w="1000" h="500" />
         <MOSFET type="P" width="wp*nfp" length="lp" nf="nfp" nrows="nrp" />
       </Cell>
       <Combine id0="MP11" id1="MP12" />
<!-- inline part of pla.xml's CellList element end -->
    </Cell>
    <Cell name="MP14" symGroupId="1" rotate="MX">
      <Box x="-300" y="0000" w="600" h="500" />
      <MOSFET type="P" width="wb*nfb" length="lb" nf="nfb" nrows="nrb"/>
    </Cell>
    ...
  </CellList>
</ Template>
```

Fig. 3.13 Part of the XML description of the layout guides (*floorplan.xml*), showing the constructs and illustrating the hierarchy, with part of the sub-floorplan file for partition P1A shown inline

no explicit definition in the template. The creation of the template is done semi-automatically from the netlist. Again the setup assistant is used to automatically create a draft with the boxes for all the cells set to (x = 0, y = 0, width = 1000, height = 1000) and the device instantiation with all the parameters.

Further automation of the template generation is possible using circuit recognition and rule extraction. Methods like the one proposed in [14] could be used to extract symmetry and grouping information, but are not considered in this work. Here, setting the symmetry constraints, the hierarchical partitioning and the positioning of the cell boxes is done manually using a plain text editor. Sub-templates can be reused automatically using a script to generate templates holding different sub-templates combinations. Multiple layout templates can be used, AIDA will select the one leading to the most compact layout. Further details on the multiple template floorplanner and layout related measures in the optimization are described in Chap. 6 of this book. At this point, the setup is complete and the circuit is ready to be designed.

3.3.3 *Graphical User Interface*

AIDA-C interface with the designer is done through a simple graphical user interface (GUI). The GUI is implemented in Java™ 1.6 providing a simple and fast way for the designer to set the target specifications and control circuit sizing optimization. The main AIDA GUI, is illustrated in Fig. 3.14.

After loading the circuit setup, the objectives, constraints and variable ranges must be set. Optionally, the settings can be loaded from a configuration file, or can

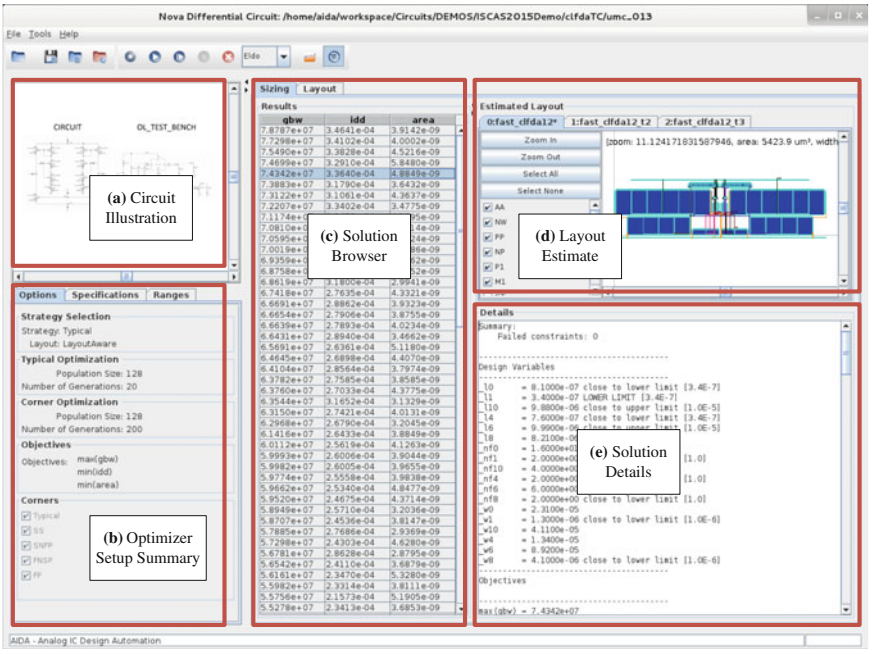


Fig. 3.14 AIDA GUI—Main panel

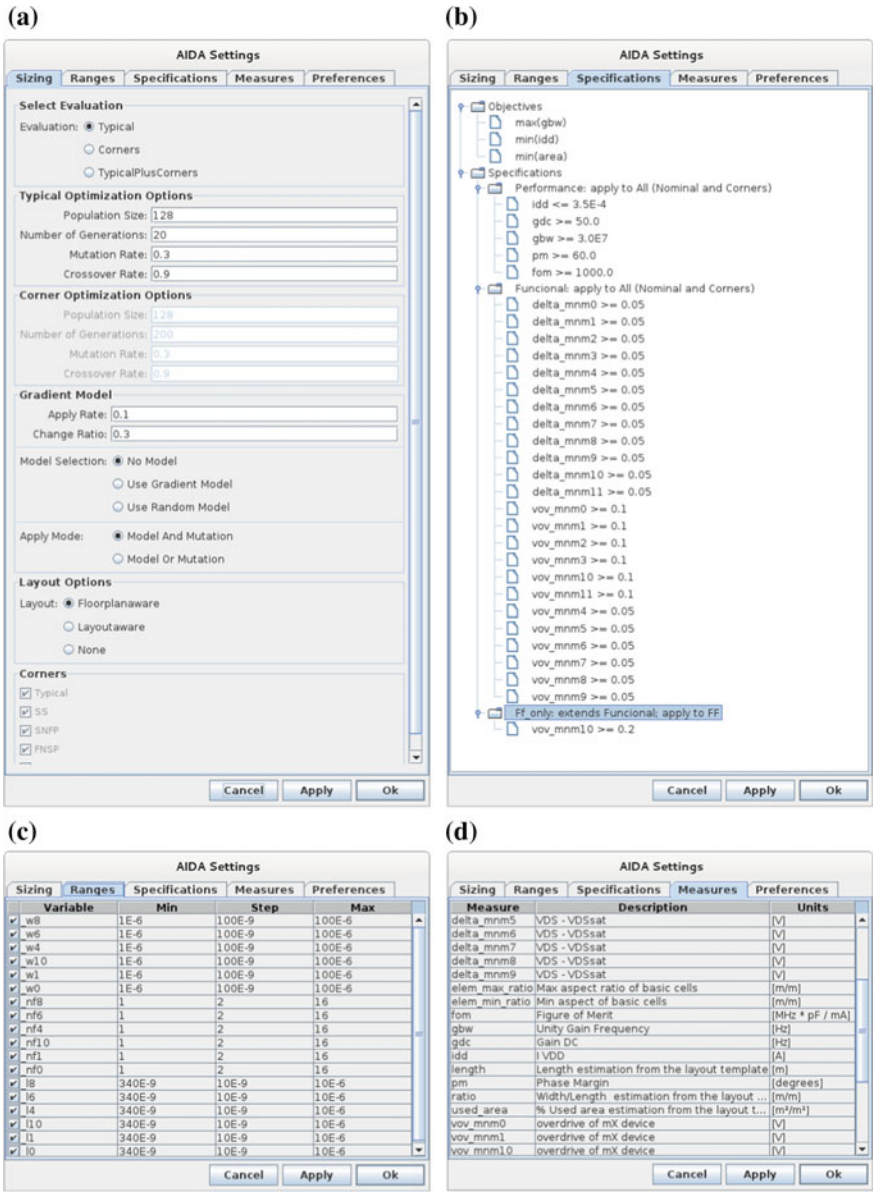


Fig. 3.15 AIDA Optimizer setup controls. **a** Sizing settings. **b** Objectives and constraints. **c** Variable ranges. **d** Measures

be stored and used later. The designer set/edit the appropriate values in the AIDA Settings dialog, which is shown in Fig. 3.15. Figure 3.15a shows the interface to set the optimizer settings, where the designer can set:

- **Strategy Selection** Select between nominal and worst case approaches.
  - **Typical** The optimization is run using only the typical testbenches;
  - **Corners** The optimization is run using only the corner testbenches;
  - **Typical + Corners** The optimization is run first using typical testbenches. When the first optimization is completed, the results are uses as a starting point for the corner optimization.
- **Typical/Corner Optimization Options** Control the number of simulations, which is the Population Size times the Number of Generations.
  - **Population Size** The numbers of elements that are evaluated at each generation, larger populations allow a better exploration of the search space;
  - **Number of Generations** The number of iterations of the algorithm, larger number of generations allows the elements to evolve better;
  - **Mutation Rate (Advanced)** The mutation rate of the evolutionary optimization kernel;
  - **Crossover Rate (Advanced)** The crossover rate of the evolutionary optimization kernel.
- **Gradient Model** Control the usage of the gradient model there are 3 options, no model, use the model, or use random generated rules
  - **Apply Rate** The % rate of application of the model;
  - **Change Rate** The % of change in the variables when the model is used.
- **Layout Options** Select the layout effects, when layout guides are defined.
  - **FloorplanAware** Only layout geometric properties are evaluated (Placer);
  - **LayoutAware** Both geometric and parasitic devices are considered;
  - **None** Layout effects are not evaluated.
- **Corners** Shows the available corners. Adding, removing or enabling corner cases must be done by editing the file "design.xml", found in the optimization project root folder (in our example/home/user/ampop_a/design.xml), and making the proper changes to the netlist(s).

In Fig. 3.15b the interface to set the circuit specifications is presented. In this panel the designer sets the performance constraints and optimization objectives.

- **Objectives** The designer can select, from the available measures, the ones to be optimized to a maximum or minimum value. Multiple objectives may be selected simultaneously.
- **Specifications** The upper and lower bounds to any of the available measures can be set in the specifications tree. The design specifications are organized in Performance Sets that group sets of constraints. The simulations (Nominal, Corner 1, … etc.) where the Set is applied can be configured using the names of the corners defined in the design.xml and the following keywords; ALL, NOMINAL, CORNERS, and the ! (not) operator, e.g., apply to ALL !

NOMINAL is the same as apply to CORNERS only. Performance Sets can extend other Sets, when extending a performance set, the new set imports all the constraints and applies them to its designated simulations, constraints can also be overridden.

Figure 3.15c shows the panel for setting the ranges of the design variables. The designer can edit the maximum, minimum and step/grid values, and also, variables can be enabled/disabled. Enabled variables are changed during optimization while disabled are not (only recombined between the solutions). One mouse right-click in the table will invert selection, while double right-click will enable all. However adding or removing design variables cannot be done from the GUI interface. It is possible to add or remove design variables by editing the file "design.xml", found in the optimization project root folder and making the proper changes to the netlist(s). Figure 3.15d shows the available circuit measures. These description are introduced when adding the circuit to the library, and cannot be edited from the interface. Again, it is possible to edit these values by editing the file "design.xml". This is how it is possible to add measures to an optimization project, after placing the appropriate measure statement in the testbench(s), when they were not defined when the circuit was placed in the library.

Optionally, the designer can define initial sizing solutions and simulate them immediately. This is done using the manual design tool shown in Fig. 3.16, which allows editing the values of the design variables of the selected sizing solution. In addition, sweeps over design variables are also supported. In Fig. 3.16a the variables can be edited manually, and using the actions in Fig. 3.16e the designer can simulate immediately the solution defined by the variables in the variable table, or execute the sweep over design variables defined in the sweep setup Fig. 3.16d.

The overall results of the simulation are shown in the table of Fig. 3.16b. The solutions are shown in rows, where the ones that meet the specifications are shown over a light green background and the ones not meeting the specifications are shown over the light red background. The columns show the simulation order, the values of the parameter that was swept, the values of the measured objectives and finally, the values of measures that fail to meet the specifications in at least one of the simulations. The failed performance values are shown in red. The panel Fig. 3.16c shows the details of the solution whose corresponding row is selected in the table. *Right-Click* in the selected row will update the design variables in Fig. 3.16a to the values of the corresponding solution.

3.3.4 Sizing

With the specifications (objectives, constraints and ranges) defined, the optimizer can now be executed. The proposed flow is by definition based on homotopy, i.e., progressively add more detail to the problem, moving from a simplified model

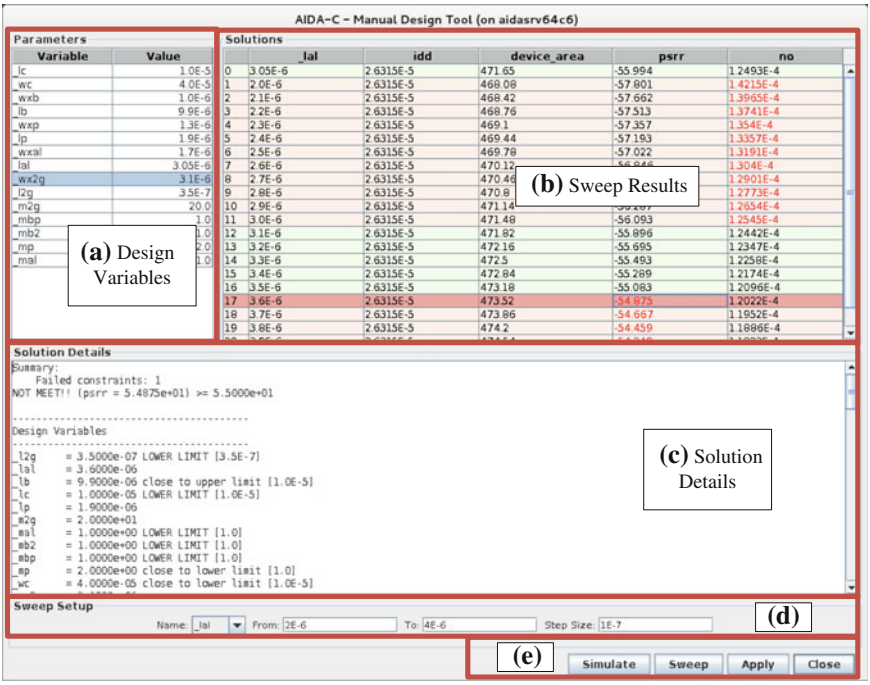


Fig. 3.16 AIDA-C—Manual edit tool

towards a more accurate (and more time-consuming) representation of the design problem, which is done by selecting the appropriate evaluation methods (e.g., nominal/worst case; no-layout/floorplan/layout). One commonly used homotopy strategy is to start with nominal simulations only, and then, use the sizing solutions obtained as the starting point for a second optimization considering the worst case corners simulations. So, besides individual optimization jobs, the nominal plus worst case corners sequence suggested is already predefined in the tool.

The optimization will use the latest results (entered manually, or loaded from a previous execution) as the starting point for the new run. When the optimizer is running, the evolution of the optimization is monitored using the plots shown in Fig. 3.17, which show an evolution of the nominal plus worst case corners optimization sequence for the two-stage Miller amplifier.

These plots are updated during the optimization and provide valuable information of its current state. In Fig. 3.17a, the *AIDA-C Convergence* plot, shows: the number of non-dominated solutions, this information useful to monitor the number of different solutions found; the sum of normalized constraint violations for the best element, which indicates how far the best solution is from complying with all the specifications, a value of zero means the solutions are feasible (i.e., all constraints verified); and finally, the dominated area, which is a relative index of the area

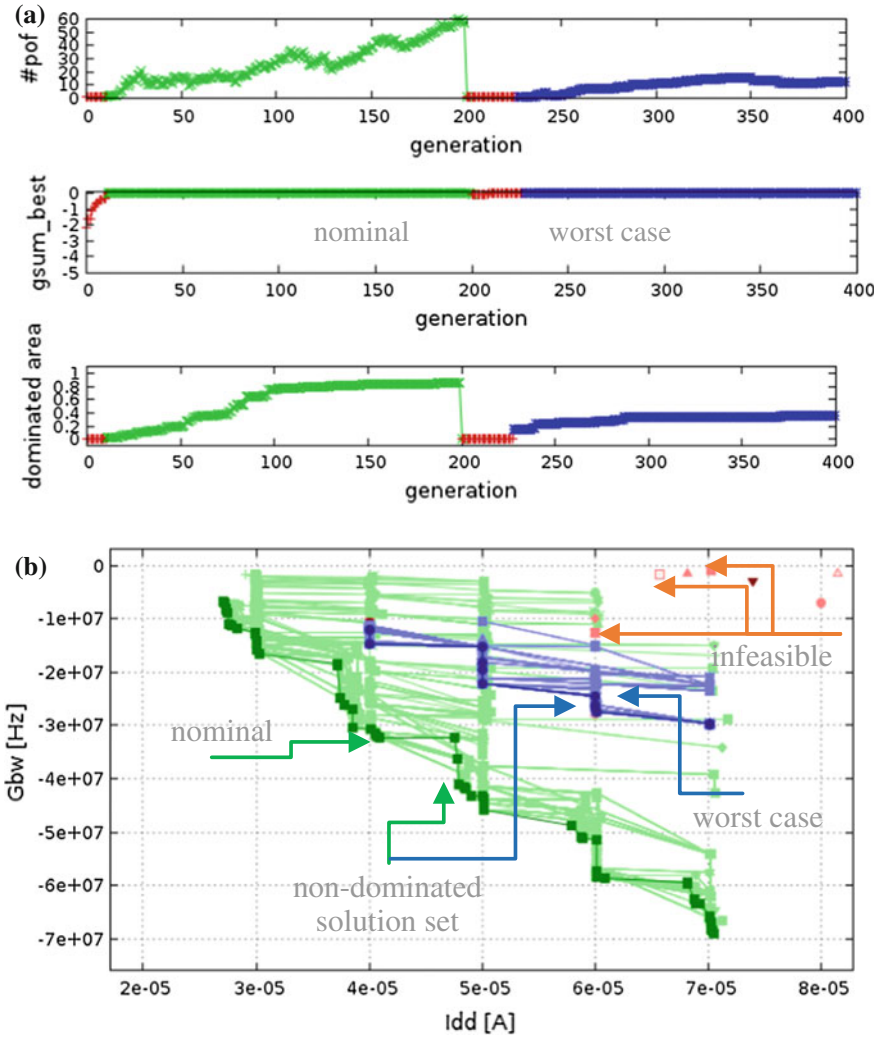


Fig. 3.17 AIDA typical plus corners monitoring plots. **a** Convergence plot. **b** Historic front plot

"above" the Pareto front. Figure 3.17b, the *Projection* plot, shows the evolution of the non-dominated fronts during the execution of the algorithm, the set of solutions is plotted at each generation. Red solutions are infeasible, green and blue are feasible, were green is used to represent a nominal strategy and blue when the corner strategy is used. The execution can be terminated before the generation limit is reached, if the solutions found are good enough, or if the designer wishes to make any adjustments.

AIDA-C's output is a family of Pareto optimal circuits that fulfill all the constraints and represent the feasible tradeoffs between the different optimization

objectives. When the optimization is completed, the obtained solutions are shown in the solution browser. The *Solution Browser* pane (Fig. 3.14c) allows the navigation through the different solutions and when one is selected, the values of the design variables, objectives and constraints of that solution are shown in the *Solution Detail* panel, as illustrated in Fig. 3.14e. If the layout guides are defined, the floorplan is shown in Fig. 3.14d. The solutions and configurations can be stored in a file for later use.

At any stage, the designer evaluates the merit of the obtained solutions and either moves to the physical design, the next step in the design flow, or re-iterates the sizing process. As AIDA-C is integrated with the layout tool these are not necessarily disjoint steps. A common iteration is to redefine the circuit to include extra design variables that are mostly related to the parametric device generators used in the layout generation, before launching the layout-aware optimization.

3.3.5 Reuse

The adopted design flow is mostly technology-independent. The setup can easily be reused either for different specifications or technologies. The most troublesome issue is when none of the circuit topologies whose setup is already defined can meet specifications. In this case, a new (or some variation of a previously defined) topology needs to be added.

Nevertheless, most of the setup (testbench(es), measures, etc.) still holds and AIDA-C can import settings between projects with different setups, matching the previously defined specifications (design variables, objectives and constraints), leaving only the newly defined variables and measures to be set. To illustrate the reusability, let us include a resistor in series with the capacitor in the two-stage amplifier. The changes in the setup are simple, just add the device in the netlist and add the new design variables l_r and w_r to the design.xml file. In terms of circuit sizing, two design strategies can be followed. A strategy is to load both configuration (ranges, objectives, constraints, etc.) and solutions, where the new variables values, which are missing in the loaded solutions, are set to the center of their range, and continue to optimize those solutions. Other option is to reuse only the previous configurations, and optimize the circuit from scratch. Figure 3.18 shows the evolution of first approaches, and Fig. 3.19 the evolution of the second.

The first gets solutions faster, i.e., in less generations, but risks sub-optimality, as it is biased toward the previous solutions. The second takes advantage of the previous setup but does not reuse the solutions avoiding the risk of getting stuck in sub-optimal solutions, but requires more computation to arrive to solutions with performance similar of the previous approach. In terms of the new topology's performance, it can be seen that the fronts for the nominal case did not shown significant changes, but when considering corners the improvement is clear, showing a wider POF when comparing to the corner POF of Fig. 3.17.

Another example of reuse is technology migration. To migrate the setup from UMC 130 nm to XFAB 350 nm technology design kit, the only changes in the

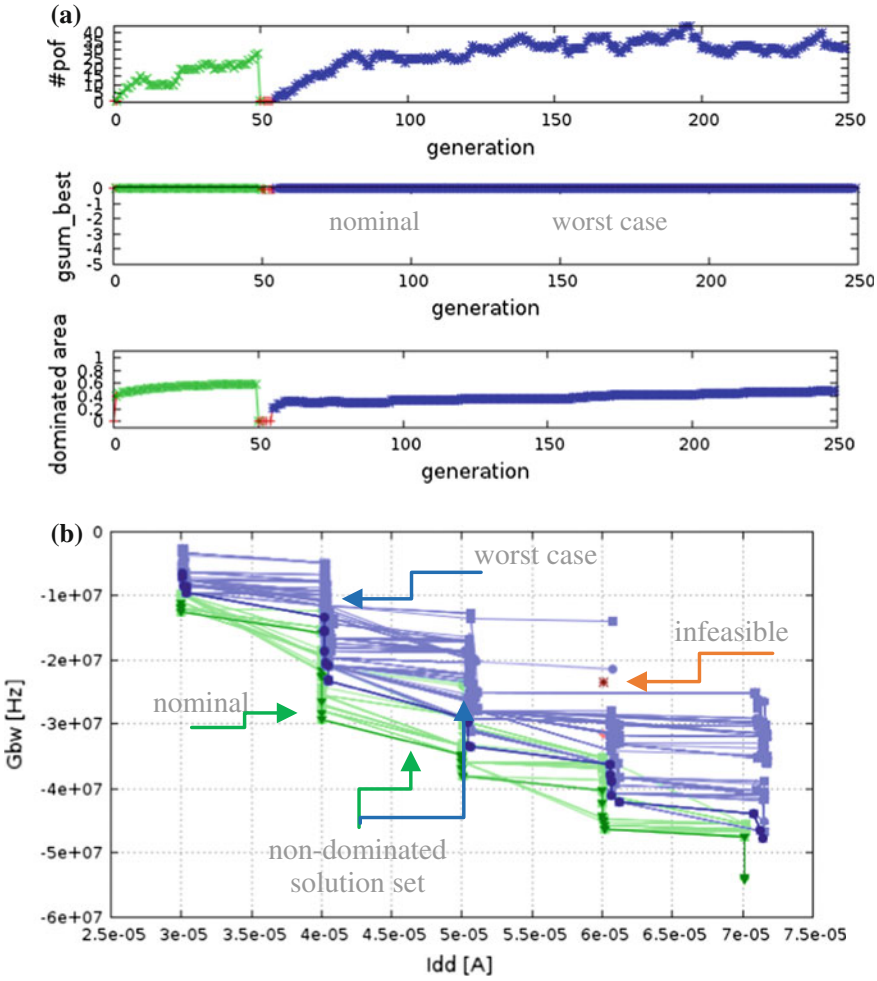


Fig. 3.18 Reuse: changing the topology and reusing the previous solutions as starting point. **a** Convergence plot. **b** Historic front plot

netlist are the device models and the voltage supply, since the topology and the measures are the same. In the new technology the variable ranges are obviously different, as well as some specifications. In this case, the specifications related to input and output swing and also the values for the overdrive and saturation margin, need to be set. In terms of sizing strategy, the only option is to optimize from scratch, as it makes no sense using sizes obtained from relatively different technology nodes (i.e., 130 and 350 nm). Figure 3.20 illustrates the changes in the netlist, and Fig. 3.21 summarizes a fast circuit optimization.

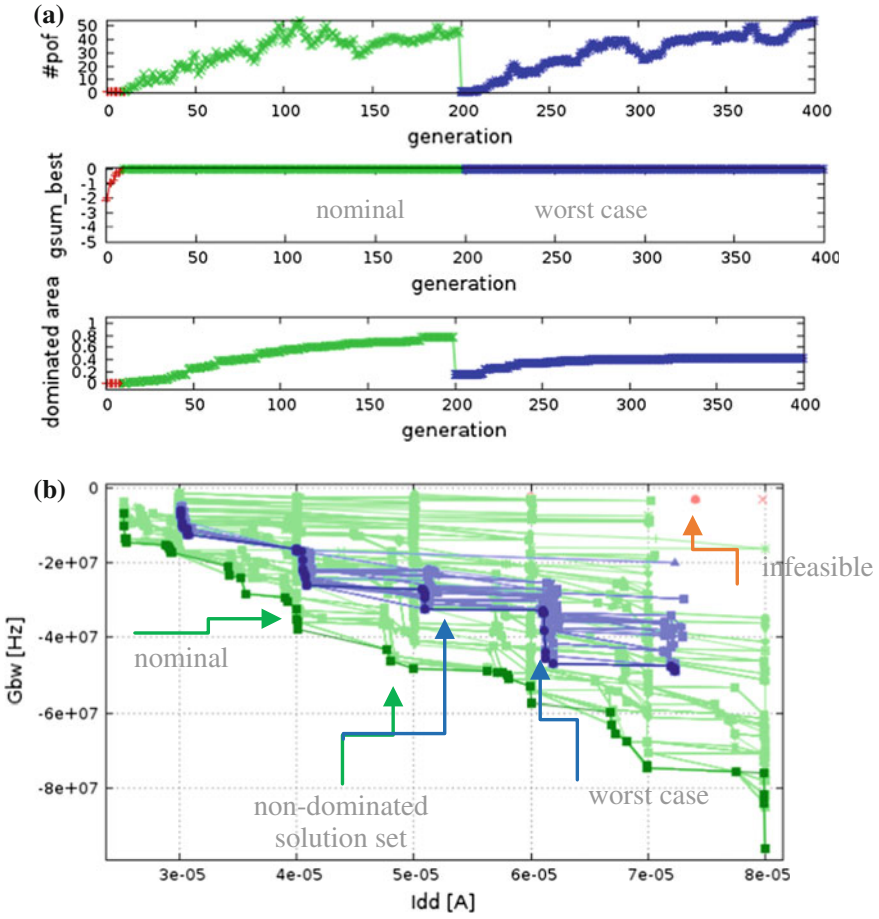


Fig. 3.19 Reuse: changing the topology and (re)sizing from scratch. **a** Convergence plot. **b** Historic front plot

```
*** Cell name: opamp_a
.SUBCKT OPAMP_A DD IN IP OUT REF SS
  XC00 OUT D12 SS        cdmm area='_wc*_lc' peri='2*(_wc+_lc)'
  MP20 REF REF DD DD     pmosi_ELT w=_wb l=_lb  m= 2
  MP14 NETZ52 REF DD DD  pmosi_ELT w=_wb l=_lb  m=_mbp
  MP22 OUT REF DD DD     pmosi_ELT w=_wb l=_lb  m=_mb2
  MP11 D11 IN NETZ52 DD  pmosi_ELT w=_wp l=_lp  m=_mp
  MP12 D12 IP NETZ52 DD  pmosi_ELT w=_wp l=_lp  m=_mp
  MN9 D11 D11 SS SS      nmosi_ELT w=_wal l=_lal  m=_mal
  MN10 D12 D11 SS SS     nmosi_ELT w=_wal l=_lal  m=_mal
  MN21 OUT D12 SS SS     nmosi_ELT w=_w2g l=_l2g  m=_m2g
.ENDS***
End of subcircuit definition.
```

Fig. 3.20 Reuse: moving to another technology

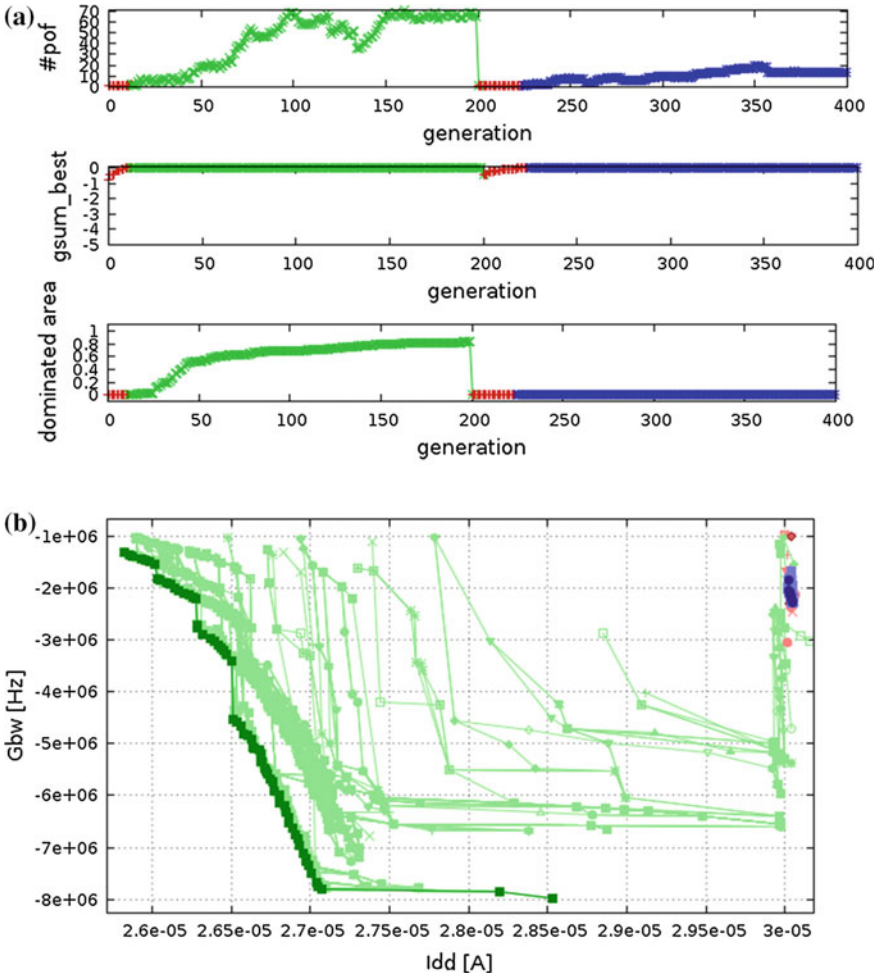


Fig. 3.21 Reuse: (re)sizing in another technology. **a** Convergence plot. **b** Historic Front plot

3.4 Conclusions

In this chapter the AIDA-C tool architecture was presented, which aims at reducing the design time of analog ICs while producing robust results. The inclusion of extreme variations by considering PVT corners analysis and the usage of the industrial circuit simulators results in an automatic circuit sizing flow that guarantees robustness. Moreover, the usage of state-of-the-art multi-objective multi-constraint evolutionary techniques enables the efficient search of the huge design space.

The outputs of AIDA-C are a set of optimal feasible solutions allowing the exploration of design tradeoffs, which eases the design task at system level. The inclusion of layout characteristics in the sizing flow reduces the number of sizing/layout iterations, and, enables an efficient fully automatic flow from sizing to layout with the integration of AIDA-C and AIDA-L. The developed flow, supported by a sturdy interface with the designer, was shown. Unlike most automation approaches that focus only in the optimization process, AIDA-C is not intended to be used as a black-box approach, but as a practical tool, allowing the interaction with the designer, while the optimizer is solving the circuit sizing problem. In addition, setup aids are introduced to further increase design automation in a flow that is highly reusable, and that embeds naturally with traditional manual sizing.

References

1. Martins R, Lourenço N, Rodrigues S, Guilherme J, Horta N (2012) AIDA: automated analog IC design flow from circuit level to layout. In: International conference on synthesis, modeling, analysis and simulation methods and applications to circuit design (SMACD), Seville, 19–21 Sept 2012
2. Lourenco N, Martins R, Horta N (2015) Layout-aware sizing of analog ICs using floorplan and routing estimates for parasitic extraction. In: 2015 Design, automation and test in Europe conference and exhibition (DATE), Grenoble, 9–13 March 2015
3. Martins R, Lourenco N, Canelas A, Povoa R, Horta N (2015) AIDA: Robust layout-aware synthesis of analog ICs including sizing and layout generation. In: International conference on synthesis, modeling, analysis and simulation methods and applications to circuit design (SMACD), Istanbul, 7–9 Sept 2015
4. Martins R, Lourenço N, Horta N (2013) LAYGEN II—Automatic layout generation of analog integrated circuits. IEEE Trans Comput Aided Des 32(11):1641–1654
5. Unutulmaz A, Dündar G, Fernández FV (2014) Template coding with LDS and applications of LDS in EDA. Analog Integr Circ Sig Process 78(1):137–151
6. Graphics M (2016) IC Verification and signoff using calibre. http://www.mentor.com/products/ic_nanometer_design/verification-signoff/. Accessed 14 April 2015
7. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
8. Bandyopadhyay S, Saha S (2013) Some Single- and multiobjective optimization techniques. Unsupervised classification. Springer, Berlin, Heidelberg, pp 17–58
9. Reyes-Sierra M, Coello Coello (2006) Multi-objective particle swarm optimizers: a survey of the state-of-the-art. Int J Comput Intell Res 2(3):287–308
10. Rocha F, Lourenço N, Póvoa R, Martins R, Horta N (2014) A New metaheuristc combining gradient models with NSGA-II to enhance analog IC synthesis. In: 2013 IEEE congress on evolutionary computation, Cancun, 2013
11. Montgomery DC (2001) Design and Analysis of Experiments, 5th edn. John Wiley and Sons, New York
12. Graphics Mentor (2013) Eldo user's manual. Release 13:1
13. Synopsys (2012) HSPICE® User guide: basic simulation and analysis, Version G-2012.06-SP2
14. Massier T, Graeb H, Schlichtmann U (2008) The sizing rules method for CMOS and bipolar analog integrated circuit synthesis. IEEE Trans Comput Aided Des Integr Circuits Syst 27 (12):2209–2222

Chapter 4
Multi-objective Optimization Kernel

4.1 Circuit Sizing as Multi-objective Optimization Problem

AIDA-C's optimization kernel implements multiple optimization algorithms, all designed to solve the general multi-objective multi-constraint optimization problem defined as:

$$\begin{array}{ll} \text{find } x \text{ that minimize } f_m(x) & m = 1, 2, \ldots M \\ \text{subject to } g_j(x) \geq 0 & j = 1, 2, \ldots J \\ x_i^L \leq x_i \leq x_i^U & i = 1, 2, \ldots N \end{array} \tag{4.1}$$

where, x is a vector of N design variables, $g_j(x)$ one of the J constraints and $f_m(x)$ one of the M objective functions. The number of design variables defines the space order, while the variable ranges will define the size of the search space. The definition in (4.1) eases the introduction of new optimization methods, and is used to create an abstraction layer between the optimization method and the circuit being optimized. However, analog designs are not specified in this manner.

Usually the analog design specifications are formulated as shown in (4.2), as a set of L, P_j^L, lower bounds and U, P_j^U, upper bounds for some of the A measured circuit characteristics, $p_i(x)$, e.g., Slew Rate, DC Gain, Settling Time, Rise Time, etc. [1]. The value of these characteristics is obtained in AIDA-C using the circuit simulator and a corresponding testbench, or, the layout generator for layout-related data, such as area, width, length, etc.

In addition, for some of the circuit's performances, like power consumption, area, speed, etc., having the minimum or maximum possible value is also specified. Hence, for an easy integration in the usual design flow and a clear definition of what is optimized and how, the design specifications, as defined by the analog designer, need to be converted into the multi-objective multi constraint formulation from (4.1) using a well-defined method [1].

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0_4

$$\begin{aligned} g_j^L &= p_i(x) \geq P_j^L \quad j = 1, 2, \ldots, L,\ i \in \{1,\ 2,\ \ldots A\} \\ g_j^U &= p_i(x) \leq P_j^U \quad j = 1, 2, \ldots, U,\ i \in \{1,\ 2,\ \ldots A\} \end{aligned} \tag{4.2}$$

The method used in AIDA-C is to consider the circuit performances being minimized directly as the objective function, $f_m(x)$, and the ones being maximized are multiplied by -1. The design specifications are normalized using their bounds, and multiplied by -1 if necessary, defining the objectives and constraints according to (4.3),

$$\begin{aligned} f_m(x) &= \begin{cases} p_m(x) & \text{when minimizing } p_m \\ -p_m(x) & \text{when maximazing } p_m \end{cases} \\ g_j(x) &= \begin{cases} \frac{p_i(x) - P_j^L}{\left|P_j^L\right|} & \text{when the design specification is } p_i(x) \geq P_j^L \\ p_i(x) & \text{when the design specification is } p_i(x) \geq 0 \\ -p_i(x) & \text{when the design specification is } p_i(x) \leq 0 \\ \frac{P_j^U - p_i(x)}{\left|P_j^U\right|} & \text{when the design specification is } p_i(x) \leq P_j^U \end{cases} \end{aligned} \tag{4.3}$$

The simple differential amplifier from Fig. 4.1 illustrates the procedure used to map the designer's specifications into the standard multi-objective optimization problem formulation from (4.1). Tables 4.1 and 4.2 show the circuit parameters, design objectives and target specifications respectively, as commonly set by the analog designer. Table 4.3 illustrates the correspondent objective and constraint functions that are considered in the optimization.

The previously introduced method to model the IC sizing problem can be used only for nominal sizing and optimization, i.e., does not account for variability effects as the circuit is evaluated using only nominal conditions. Despite the output does not consider the limitations imposed by variations of process and environment parameters, it is still useful for the circuit designer to perform trade-off analysis and speed the search for feasible solutions, as it requires less circuit simulations (execution time) to estimate the performance of the tentative solutions.

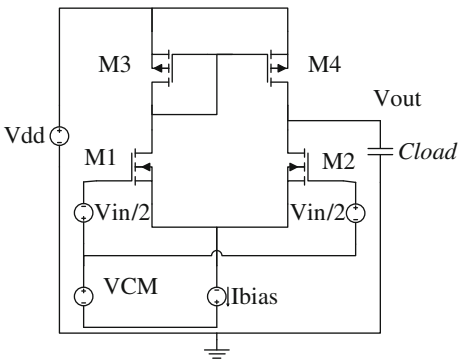


Fig. 4.1 Schematic of the simple differential amplifier and testbench

Table 4.1 Parameters ranges for the differential amplifier example

Var.	W_{12} (µm)	W_{34} (µm)	L_{12} (µm)	L34 (µm)	Ib (µm)
Max.	2000.0	2000.0	10.0	10.0	100
Min.	0.240	0.240	0.120	0.120	10

The variables L_{12} and W_{12} are dimensions of M1 and M2 transistors and L_{34} and W_{34} of M3 and M4 transistors

Table 4.2 Objectives and specifications for the differential amplifier example as commonly defined by the analog designer

	Performance	Target	Units	Description
Objectives	Gain	Maximize	dB	DC gain
	Gbw	Maximize	MHz	Unit-gain frequency
Constraints	Gbw	≥ 50	MHz	Unit-gain frequency
	Pm	$55 \leq$ pm ≤ 90	°	Phase margin
	vov_m1, vov_m2	≥ 50	mV	Vgs − Vt
	vov_m3, vov_m4	≥ 100	mV	Vgs − Vt
	delta_m1, delta_m2	≥ 50	mV	Vds − Vdsat
	delta_m3, delta_m4	≥ 50	mV	Vdsat − Vds

Table 4.3 $f_m(x)$ and $g_j(x)$ for the differential amplifier example

Objectives	$f_0(x) = -gain$	$f_1(x) = -gbw$	
Constraints	$g_0(x) = \frac{gbw}{50\times10^6} - 1$	$g_1(x) = \frac{pm}{55} - 1$	$g_2(x) = 1 - \frac{pm}{90}$
	$g_3(x) = \frac{vov\_m1}{50\times10^{-3}} - 1$	$\cdots$	$g_{15}(x) = \frac{delta\_m4}{50\times10^{-3}} - 1$

However, IC fabrication is not a deterministic process, and solution robustness is of the utmost importance in this industry. For a robust IC design that ensures that the vast majority of the fabricated circuits will work according to the specifications when exposed to variability in both fabrication process and environment, and as seen in Chap. 2 of this book, worst case parameters sets or worst case corners are usually considered [2–7]. Worst case parameters sets are sets of values for the environment and process parameters, which are not controlled by the designer, that lead to the worst values of circuit's performances.

Therefore, the IC sizing optimization problem must be redefined with the original design variables and some additional parameters, as described in Eq. (4.4), where, x is a vector of N optimization variables as in (4.1), and δ a vector that represents the random process and environment variations.

$$\begin{array}{ll} \text{find } x \text{ that minimize } f_m(x,\delta) & m = 1, 2, \ldots M \\ \text{subject to } g_j(x,\delta) \geq 0 & j = 1, 2, \ldots J \\ x_i^L \leq x_i \leq x_i^U & i = 1, 2, \ldots N \end{array} \tag{4.4}$$

AIDA-C takes a worst case approach where the design is evaluated at all the worst-case corners defined by the designer. This increases in the number of simulations for each evaluation makes the execution slower when compared to nominal design, but the output circuits are ensured to be feasible in all tested corner conditions.

During optimization, each tentative circuit is simulated over C, the number of corners being considered, different instances of the random variables, δ_c, and AIDA-C considers the worst performance obtained in those multiple corners. In this worst case approach, each objective, which is being minimized, is evaluated to the maximum value obtained from the simulations of the circuit in all corners, and also, each constraint is evaluated by adding its contribution in all the corner cases where it is violated, leading to the modified objective and constraint Eq. (4.5). Where, $f_m^c(x,\delta_c)$ and $g_j^c(x,\delta_c)$ are, respectively, the objective $f_m(x)$ and the constraint $g_j(x)$, as defined for the nominal case, but considering the circuit's performance obtained from C different simulation conditions.

$$\begin{aligned} \hat{f}_m(x) &= \max_{c=1,2,..C} \left(f_m^c(x,\delta_c)\right) \\ \hat{g}_j(x) &= \sum_{c=1}^{C} c_j^c(x,\delta_c) \text{ with } c_j^c(x,\delta_c) = \begin{cases} 0 \text{ if } g_j^c(x,\delta_c) \geq 0 \\ g_j^c(x,\delta_c) \text{ if } g_j^c(x,\delta_c) < 0 \end{cases} \end{aligned} \tag{4.5}$$

Given the maturity of the analog IC design flow with respect to worst case design, enumeration by the designer is a reasonable approach to define the worst case corners that are used in the optimization loop. Also by using user-defined corner cases, the proposed methodology goes towards the need of some design-houses where a user defined corner signoff is mandatory.

Nevertheless, and as surveyed in Chap. 2 of this book, fully automatic solution to determine the worst case corners are available. Methods such as 3–sigma corners or Worst Case Distances could be used to automatically determine the worst case corners. In both techniques the worst case corners are obtained by finding, in two distinct manners, the values of δ that minimizes the circuit's performances with respect to each of the objectives and constraints.

4.2 Optimization Kernel

With the analog IC sizing and optimization problem defined, the next sections discuss the implemented optimization methods. Since both nominal and worst case optimization problems are mapped to the tuple (x, f, g) there is no distinction from

optimization method perspective, which eases the implementation and introduction of new methods. Besides, handling multiple methods to measure the circuit's performance is also facilitated.

The multi-objective optimization kernels that are implemented in AIDA-C optimizer, illustrated in Fig. 4.2, are the NSGA-II [8], MOPSO [9] and MOSA [10]. Additionally, the individual algorithm were implemented having in mind the possibility of intermingling the optimization strategies in some hybrid methods.

Having multiple metaheuristics available opens the opportunity to explore their strong points while trying to overcome their shortcomings. For example, in NSGA-II the crossover is good to explore the diversity in the population, though, the mutation capability for local improvement is reduced as it does not allow for the fitness of a solution to deteriorate, as simulated annealing (SA) allows. On the other hand, convergence to a better solution is slower using crossover than following the leader in the particle swarm optimization (PSO) strategies, however, the risk of getting stuck in local minima is larger in the last. Some flavours of these different trade-off between exploration and exploitation are tested with the implementation of hybrid multi-kernel solutions.

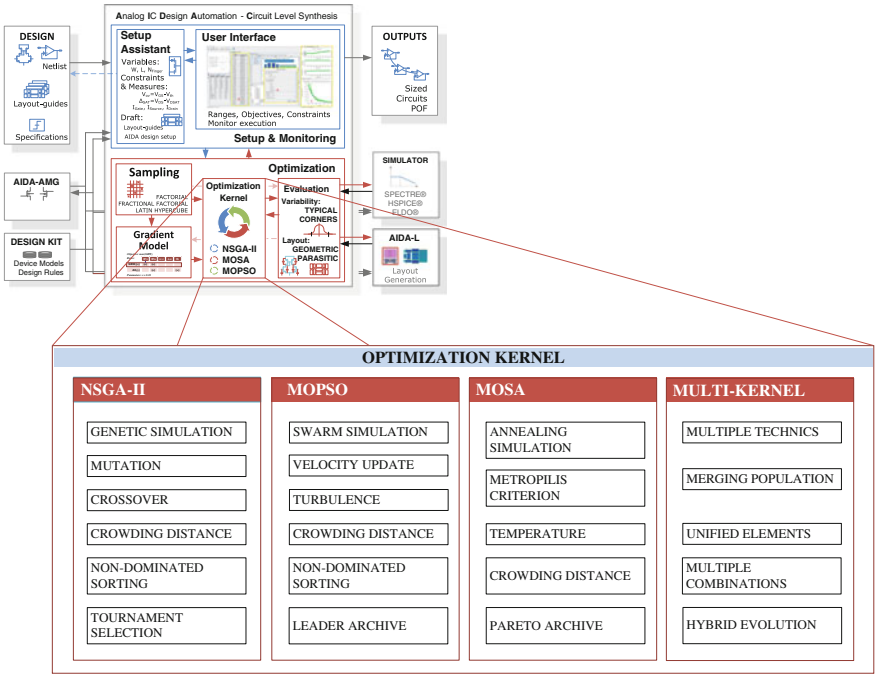


Fig. 4.2 Algorithms implemented in AIDA-C's optimization kernel

4.2.1 NSGA-II

The NSGA-II [8] kernel is an evolutionary optimization scheme, which means that it mimics, to some extent, natural evolution. The genetic algorithm starts by generating an initial population of individuals, each one representing a different solution, i.e., the initial parents. To ensure a good exploration of the search space, this first population must offer a wide diversity of genetic material. Generally, the initial population is generated randomly but other sampling methods can be used, as discussed in Sect. 4.3.1 of this chapter.

New individuals are obtained from the current population by the application of the genetic operators: mutation and crossover. The crossover operation uses two population elements, called parents, to generate the new elements, called offspring, combining randomly selected sets of information from each of the parents into the offspring. The mutation is a random change in one individual's genetic information. The mutation operator introduces new information which helps escaping from local minima and increases the diversity in the population, whereas, the crossover recombines pieces of information already present in the population.

The new individuals' fitness, which correspond to an evaluation of how good the candidate solution is, is computed and they are ranked together with the parents. The fittest individuals are selected as the new parents, and the less fit discarded. The process is repeated until the convergence or ending criterion is reached. The distinguishing characteristic of NSGA-II is that the ranking is made by using Pareto dominance to sort the multi-objective solutions. The pseudo code of NSGA-II is shown in Algorithm 4.1.

Algorithm 4.1 NSGA-II Procedure [8]

1. Set $t=0$. create a random population R_0 of size $2N$.
2. Evaluate the initial population, R_0.
3. Using the fast non-dominated sorting algorithm, identify the non-dominated fronts F_1, F_2, …, F_k in R_t.
4. For $i=1,\ldots,k$ do:
5. Calculate crowding distance of the solutions in F_i.
6. Create P_{t+1} as follows:
   If $|Pt_{+1}|+|Fi| \leqslant N$, then set $P_{t+1}=P_{t+1} \cup F_i$;
   If $|Pt_{+1}|+|Fi|>N$, then add the least crowded $N-|P_{t+1}|$ solutions from F_i to P_{t+1}.
7. If the stopping criterion is satisfied, stop and return F_1.
8. Use binary tournament selection based on the crowding distance to select parents from P_{t+1}.
9. Apply crossover and mutation to P_{t+1} to create offspring population Q_{t+1} of size N.
10. Evaluate offspring population Qt.
11. Set $R_{t+1}=P_{t+1} \cup Q_{t+1}$
12. Set $t=t+1$, and go to 3.

Pareto dominance, which states that one point in the solution space, A, is not dominated by another point, B, if $\exists m: f_m(\mathrm{A}) < f_m(\mathrm{B})$, is considered to sort the solutions by rank, where the rank of the individuals is set by finding the non-dominated fronts iteratively. The rank 1 individual are the ones that are not dominated by any other. These individuals are then removed from the population and the process is repeated for the next ranks until there are no more individuals in the population. The individual with lower rank dominate the ones with higher rank. Algorithm 4.2 illustrates the procedure.

Algorithm 4.2 Non-dominated sorting procedure for a population P of size N

1. Set $F_1=\emptyset$ and $i=1$
2. While P is not empty
3. For $j=1,\ldots,|P|$ do:
4. For $k=1,\ldots,|P|$ and $j!=k$, do:
5. if $P[j]$ is dominated by $P[k]$ then continue loop in j.
6. $F_i = F_i+\{P[j]\}$
7. Remove F_i from P
8. set $i=i+1$ and go to 2

This cubic approach to compute the rank of the elements in the population was introduced to clearly describe the non-dominated sorting, however, the fast non-dominated sorting algorithm described in [8] is used. It first computes the dominance between all solutions, storing the set of elements that are dominated (Sd) and the number of elements that dominate (d) for each solution. Using this information, all the solutions that are not dominated (have zero elements dominating them, i.e., $d = 0$) have their rank set to 1 and are removed from the elements' pool, decrementing d for all the solutions in their Sds. The process is then repeated for rank 2, and successively until the elements' pool is empty, leading to a quadratic algorithm.

To solve ties between elements of the same rank, the crowding distance is used. The crowding distance is an estimate of the density of elements. Each element with the same rank is assigned a value that measures to the distance to the closer elements. Figure 4.3 illustrates four ranking fronts and shows graphically the components used to compute the crowding distance of solution B in a problem with two objectives.

The crowding distance of the elements in a front is computed by iterating in the M objective functions, sorting the elements using each objective and for each element accumulating the normalized value of the distance between the elements

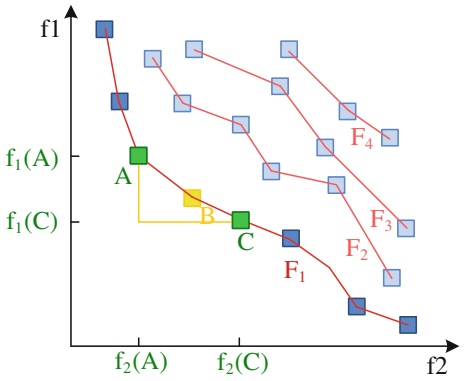


Fig. 4.3 Fronts for multiple ranks, and crowding distance illustration for solution B

before and after in the ordered set. The boundary elements (element with smaller and higher value of each objective) are assigned with infinite value of crowding distance. The pseudo code of the crowding distance computation is shown in Algorithm 4.3.

Algorithm 4.3 Crowding distance (*cd*) computation of front F_T of size N in a problem with M objectives

1. For `i=1,…,N` do: `F_T[i].cd=0` //initialize crowding distance to 0
2. For `m=1,…,M` do:
3. `S = F_T` sorted ascending using the objective function `f[m]`
4. `S[1].cd = S[N].cd =` ∞
5. For `i=2,…,N-1` do:
6. `S[1].cd+=(S[i+1].f[m] - S[i-1].f[m])/(S[N].f[m] - S[1].f[m])`

4.2.2 MOSA

MOSA is an adaptation of the single-objective SA [11] to the multi objective case. Like evolutionary algorithms, SA is inspired by a natural phenomenon. The algorithm simulates the cooling and annealing of liquid material. When a liquid material cools and anneals quickly, the material will solidify into a sub-optimal configuration. However if the liquid material cools slowly, the crystals within the material will solidify optimally into a state of minimum energy, i.e., ground state.

The algorithm steps are outlined in Algorithm 4.4. Despite the differences of concept, when comparing SA with stochastic hill climber, one cannot ignore the similarities. Nevertheless, in the SA the cooling schedule introduces the ability to explore broader solutions in the beginning, when the temperature is higher and perform almost like hill climber for very low temperatures. This makes cooling schedule extremely important to the performance of the algorithm.

Algorithm 4.4 Single-objective simulated annealing

1. Set `T=Tmax`
2. Create a random solution *u*
3. Set `fu=cost` of `u`
4. While `T < Tmin` do:
5. Set `v`=neighbour of `u`
6. `fv = cost of (v)`
7. If `fv < fu then u=v`
8. Else if `rand(0,1) < exp((fu - fv)/(fu.T))` then `u=v`
9. Update temperature `T`
10. Return `u`

In theory, an infinitesimal decrease of T over time leads to the global optimum solution. From a practical point of view, there are some important aspects to consider when devising the cooling schedule. If the initial temperature is too low or if temperature drops too fast, premature convergence occurs. Nonetheless, if the initial temperature is too high or if temperature drops too slowly, finding the solution will take longer. This is the main trade-off when designing the cooling schedule, and one common scheduling procedure is the exponential cooling:

$$T_{K+1} = \frac{T_1}{T_0}^K T_K. \tag{4.6}$$

Since the nature of SA is to explore the neighbourhood of single tentative solutions, the adaptation to multi-objective requires changes in the structure of the search. The straightforward approach is create a weighted combination of the objectives, and find the set of Pareto optimal solutions with multiple runs of the SA with different weights, however, finding the correct weights is a complex task. Another adaptation can be found in [10], where the adaptation to the multi-objective case is done using an archive that stores the best solutions, and, the acceptance of a new solution is based on the dominance with respect to the archive, not just to the current solution. However, by exploring the neighbourhood of just one solution at a time, the diversity of the solutions found is degraded.

The MOSA implementation in this work also follows an archive-based technique, but exploring the neighbourhood of the entire archive at each iteration, instead of only a single point. This way, the diversity of the solutions explored increases. Moreover, another practical advantage is that simulating multiples circuits in batch is more computationally efficient than simulating one circuit at a time.

To increase the potential for more annealing branches, the algorithm starts with P elements in the archive instead of only one. The acceptance is still made by Metropolis criterion [12] and the archive is trimmed by non-dominated sorting and

crowding distance, when its size exceeds the maximum number of elements permitted. A Pareto set with all the non-dominated solutions found is also kept. Since the new elements are neighbours of the original solutions, the Pareto archive grows faster by taking solutions that are close to each other, and crowding distance is used to trim the Pareto set. The implemented MOSA is shown in Algorithm 4.5, instead of using $T_{\min}$ to control the annealed schedule, a maximum number of iterations is set to control the annealing.

Algorithm 4.5 Multi-objective simulated annealing

1. Set *T=Tmax* and *iteration=0*
2. Set *archive=P* random solutions
3. Evaluate the cost of all solutions in the *archive*
4. Set *pareto* = non dominated solutions in *archive*
5. While *iteration < N* do:
6. Create *neighbors* for each element in *archive*
7. Evaluate the cost of all *neighbors*
8. For each *element,neighbor u,v in archive,neighbors*
9. if *v* dominates *u* then *v* replace *u* in archive
10. else if u does not dominate v then add *v* to *archive*
11. else if rand(0,1)<exp(|f(*u*)–f(*v*)|/(|f(*u*)|.T)) then *v* replace *u* in archive
12. Set *pareto to* non-dominated solutions in union of archive and *pareto*
13. Trim *pareto* and *archive* sets using *crowding distance*
14. Update temperature *T, iteration++*
15. Return *pareto*

4.2.3 MOPSO

The PSO algorithms introduced by Kennedy and Eberhart in 1995 [13] are partly inspired by the behaviour of large animal swarms such as schooling fish, flocking birds or honey bees. PSO associates each particle to a candidate solution and let them explore the search space. This technique is focused on the collective behaviour of a distributed population of simple agents that interact locally with each other. Each particle is associated with a stochastic velocity vector which indicates where the particle is moving to. The next move of each particle at a given time, illustrated in Fig. 4.4, is a stochastic combination of: the velocity in the previous time instant; the direction towards the best position ever occupied by the particle; and, the direction towards the best swarm positions. In the standard model each particle *i* is associated with a position (x_i) in the search space, a velocity (v_i), the position (p_i), the best

location encountered by the particle, and the location of the best particle in the swarm. The interaction between these variables is governed by the following rules:

$$\begin{cases} \vec{x}_{i+1} = \vec{x}_i + \vec{v}_{i+1} \\ \vec{v}_{i+1} = w\vec{v}_i + \vec{\varphi}_{i1}(p_i - \vec{x}_i) + \vec{\varphi}_{i2}(p_g - \vec{x}_i) \\ \vec{\varphi}_{i1} = U(0, \vec{\varphi}_{1\max}) \\ \vec{\varphi}_{i2} = U(0, \vec{\varphi}_{2\max}) \end{cases} \tag{4.7}$$

where, w is the inertia weight and p_g is the position of the best particle in the swarm. The vectors φ_1 and φ_2 are randomly generated at each iteration for each particle with entries uniformly distributed between 0 and $\varphi_{1\max}$ or $\varphi_{2\max}$, respectively.

The manipulation of some of these parameters develops other variations over the standard algorithm. Controlling the velocity and the direction to the particle's best position allows the implementation of schemas for exploitation and exploration of the search space. The MOPSO implemented follows the implementation described in [9], using external archive, turbulence, and a fully connected topology and density estimator (crowding distance). In the single-objective approach, one of the critical factors in the implementation of a PSO is the selection of the leader, in the multi-objective case the issue is aggravated, as there are many options to select the leader. The chosen method was to randomly select a solution from the Pareto to increase the pressure for improvement. Other possibilities are valid, such as selecting random solutions from the non-dominated set of particles and use crowding distance tournament between two solutions in order to select the leader. The pseudo-code for the MOPSO is shown in Algorithm 4.6.

Algorithm 4.6 Multi-objective PSO

1. Set *iteration*=0
2. Set *particles* = P new random solutions
3. Evaluate all *particles*
4. Set *pareto* = non dominated solutions in *particles*
5. While *iteration* < *N* do:
6. `For each particle p in` *particles*
7. `g = select leader from` *pareto*
8. `Update p according to equation (4.4)`
9. `Add turbolence to p`
10. Reevaluate all *particles*
11. Set *pareto to* non-dominated solutions in union of *particles* `and pareto`
12. `Trim pareto set using crowding distance`
13. iteration ++
14. `return pareto`

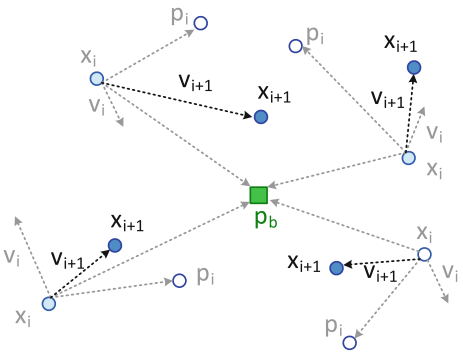


Fig. 4.4 Particle update in PSO

4.2.4 Multi-kernel Algorithms

With the algorithms previously described implemented in the tool, new methods that combine their techniques can be explored. By combining and reusing the diverse strategies it is possible to take advantage of their strong points. Therefore, the framework flexibility is further enhanced and there is the possibility to find combinations of the different methods that may outperform the individual scheme.

4.2.4.1 Parallel Multi-kernel

One possible combination is to use multiple algorithms in parallel, as shown in Algorithm 4.7, sharing the elements to solve the problem more efficiently. The parallel combination uses a pool of elements that is redistributed between each kernel and evolved using a different approach.

Algorithm 4.7 Parallel multi-kernel

```
1. Set iteration=0
2. Set archive = P new random solutions
3. Evaluate all solutions in archive
4. Set pareto = non dominated solutions in archive
5. While iteration < N do:
6.         If the merge condition is true
7.                 For each kernel k in kernels[]
8.                         redistribute-elements(k, archive)
9.         For each kernel k in kernels[]
10.                execute-kernel-step(k)
11.        Collect elements from all kernels[] to archive
12.        Set pareto to non-dominated solutions in union of archive
   and pareto
13.        Trim pareto and archive sets using crowding distance
14.        iteration++
15. return pareto
```

The major decisions in this method is when to redistribute the elements and how the redistribution is done. To redistribute the elements, a simple method is to rearrange the samples at uniform periods of time, i.e., at each fixed number of steps. This was the method implemented, nonetheless, any alternative method could be devised. Regarding the redistribution itself, the three methods illustrated in Fig. 4.5 were considered. In Fig. 4.5a is presented a simple version that shuffles and reassign the elements to the different kernels, taking advantage of the different explorations techniques. In Fig. 4.5b a more greedy approach is used, following the same principle of using different methods to explore the same space, but that selects only the best individuals using rank and crowding distance. Those best individuals are set to all the kernels. A third approach, shown in Fig. 4.5c, sorts the elements using some criteria and split them in blocks, allowing the algorithms to explore different regions of the search space.

4.2.4.2 Sequential Multi-kernel

A different method to combine the kernels is to used them sequentially, where the results from one kernel is passed as input to the next, as shown in Algorithm 4.8. With this combination is possible to optimize a problem for example using N_1 cycles of NSGA-II, and then, N_2 cycles of MOSA, for fine tuning of the solutions.

An important detail is that, like MOSA and MOPSO, both parallel and sequential multi-kernel approaches use an external Pareto set to store the best solutions found through the several iterations of the different kernels. Some of these approaches were experimented to show that the tool can take advantage of such advanced combination of kernels, but the detailed exploration of hybrid optimization techniques goes beyond the scope of this work.

Algorithm 4.8 Sequential multi kernel

1. Set *iteration*=0
2. Set *archive* = P new random solutions
3. Evaluate all solutions in *archive*
4. Set *pareto* = non dominated solutions in *archive*
5. While *iteration* < *N* do:
6. `Select kernel k from` *kernels[]*
7. `execute-kernel-step(k)`
8. `Collect elements from k to archive`
9. Set *pareto to* non-dominated solutions in union of *archive* `and pareto`
10. `Trim pareto and archive sets using crowding distance`
11. iteration++
12. `return pareto`

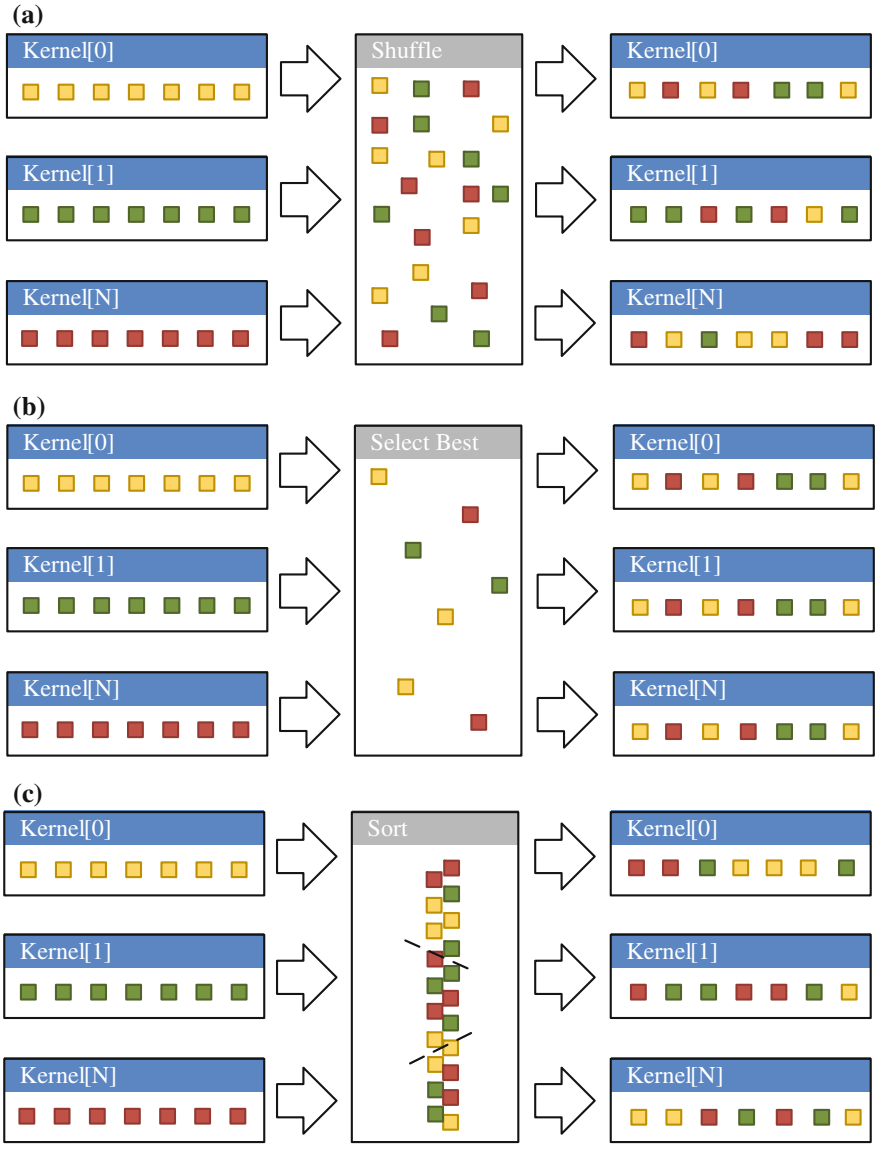


Fig. 4.5 Different methods to redistribute elements in the parallel multi-kernel approach. **a** Shuffle. **b** Best. **c** Sorted

4.3 Enhancing the Optimization with Machine Learning

To improve the performance of the optimization, machine learning techniques are used to add automatically generated design knowledge to the optimizer. A 1st order simplistic model, the Gradient Model, is used to guide the generation process to a better solution faster. The introduced simple differential amplifier will be used again throughout the next section as a working example to illustrate the implemented methodology. To simplify the illustrations, only the input variables W_{12} and W_{34}, which represent the widths of the transistors pairs (M1, M2) and (M3, M4) respectively, are considered.

The generation of the model starts by sampling the design space using a Design of Experiments (DOE) approach [14], and then, the model is generated from those samples. The samples are processed by extracting and ranking the contributions of each design variable (input) to each design performance or objective (output), and, finally, building a set of gradient rules that will be combined with the evolutionary optimization kernel.

4.3.1 Sampling the Design Space Using DOE

Two factorial DOE variants were explored within this work, namely, the full factorial design and the fractional factorial design. The number of simulations required to construct the DOE's matrix for both variants obeys Eq. (4.8),

$$\#\,simulations\, N_S = B^{(n-p)} \tag{4.8}$$

where B is the number of points per design variable or matrix base $B > 1$, n is the number of design variables, and p the number of non-elementary design variables. A non-elementary design variable is a variable that is not sampled in all possible combinations with the other variables. The value of p is zero in the case of full factorial design, and equal or greater than 1 for the fractional factorial design.

4.3.1.1 Full Factorial Design

In the full factorial DOE, the circuit is sampled in all the combinations of variables' levels. For each variable (x_i), B logic levels are defined, and to each value, it is assigned a value $v_{i,b}$ derived from the variable's range according to (4.9). In Table 4.4, the mapping between logic values and variable values is illustrated for a subset of variables (the device's widths and bias current) of the simple differential amplifier introduced in Sect. 4.1 of this chapter, for a B set to 2.

Table 4.4 Association between range values and DOE's level

Variables\Levels	0	1
$x_1 - W_{12}$ (μm)	500.180	1500.060
$x_2 - W_{34}$ (μm)	500.180	1500.060
x_3 − IBias (μA)	32.5	77.5

Table 4.5 DOE's matrix for full factorial design

	x_1	x_2	x_3	y_1 − gain (dB)	y_2 − gbw (MHz)
1	0	0	0	30.61	13.17
2	1	0	0	29.72	12.00
3	0	1	0	30.54	10.14
4	1	1	0	29.68	9.46
5	0	0	1	30.88	29.89
6	1	0	1	30.55	27.51
7	0	1	1	30.80	23.16
8	1	1	1	30.48	21.62

$$v_{i,b} = X_i^{\text{Min}} + \frac{\left(X_i^{\text{Max}} - X_i^{\text{Min}}\right)}{2B} \times (1 + 2b), \quad b = 0, \ldots, B-1 \tag{4.9}$$

Once the variables mapping is complete, the next step is to construct the DOE's matrix. The matrix has one line per each possible combination of values, being the total number of lines given by (4.8), where p is 0. The columns are the inputs *(x)*, identified with the logic levels, and the outputs *(y)* described by the measured value. The concatenation of the values in the inputs is also referred as the code of the sample, as it acts as unique identifier.

Table 4.5 shows the 8 (2^3) sample matrix, obtained for the simple differential amplifier example. If any vector produces an output that is not measurable, it is not taken into account during the generation of the model.

4.3.1.2 Fractional Factorial Design

From (4.8), the number of samples (and obviously simulations) increases exponentially with the number of input variables. To reduce this effect, the fractional factorial DOE introduces the notion of non-elementary variable, as a variable that is not used to generate the code of the sample, reducing the size of the matrix. The level of the non-elementary variables is determined from the code, i.e., from the levels of the elementary ones. Several methods to compute the level of non-elementary variables are available in the literature [14], in this work the level Ln_i of the non-elementary variable i is given by:

$$Ln_i = (L_1 + L_2) \mod B \tag{4.10}$$

Table 4.6 DOE's matrix for fractional factorial design with x_2 non-elementary

	x_1 – W_{12}	x_2. – W_{34}	x_3 – IBias	y_1 – DC gain (dB)	y_2 – GBW (MHz)
1	0	0	0	30.61	13.17
2	1	1	0	29.68	9.46
3	0	1	1	30.80	23.16
4	1	0	1	30.55	27.51

Table 4.7 DOE's matrix for fractional factorial design with x_1 non-elementary

	x_1 – W_{12}	x_2. – W_{34}	x_3 – IBias	y_1 – DC Gain (dB)	y_2 – GBW (MHz)
1	0	0	0	30.61	13.17
2	1	1	0	29.68	9.46
3	1	0	1	30.55	27.51
4	0	1	1	30.80	23.16

where *mod* is the modulo operator and, L_1 and L_2 are the levels of the first and second elementary variables, respectively, which ensures an even distribution in the levels. Table 4.6 shows the 4 (2^{3-1}) sample matrix obtained for the simple differential amplifier example, by considering the variable x_2 as a non-elementary variable, while Table 4.7 shows the same but considering x_1 as the non-elementary variable. It is easy to see that the tables are the same with the simulation out of order.

4.3.1.3 Latin-Hypercube Design

In the Latin hypercube design, the circuit is sampled such that there is only one point at each level. Hence, the number of simulations is B, the number of logic levels for the variables, regardless the number of variables. This is one of the benefits of the Latin hypercube, as the number of samples does not depend on the number of variables, but only on the number of levels, as it can be easily controlled. However, this does not mean that Latin hypercube design does not depend of the number of variables. Like all other sampling methods, the Latin hypercube design does suffer from the curse of dimensionality, and the space filling degrades exponentially with the number of variables. Other interesting property is that removing variables from a Latin hypercube design results in a (eventually sub-optimal) Latin hypercube design.

A Latin hypercube design is not unique, for B levels and n variables there are $(B!)^{n-1}$ designs. Originally Latin hypercube designs were constructed by sequentially generated random samples that meet the Latin hypercube design criteria. However not all designs have equally good space-filling properties, as shown in Fig. 4.6.

To ensure good space filling and or column-wise orthogonally several methods to derive good Latin Hypercube designs have been proposed [15–18]. Space-filling

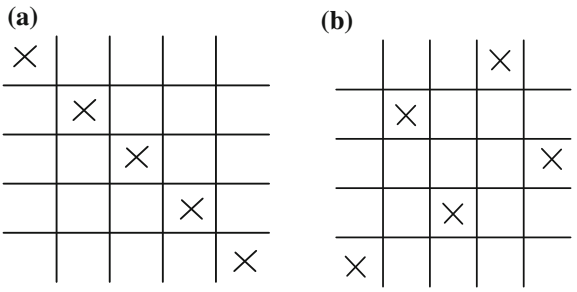


Fig. 4.6 Latin hypercube design with 2 variables and 5 levels. **a** Design with poor space-filling. **b** Design with good space-filling

is improved by minimizing some form of distance metric, whereas orthogonality may be obtained by considering column-wise correlations.

Given the flexibility of the Latin hypercube design, in AIDA-C the Latin hypercube sampling (LHS) is done using pre-optimized designs. A set of small designs optimized for the number of variables ranging from 22 to 64 with less than 64 samples from [15] is used. The designs are stored in a file and used whenever needed. Whenever sampling a circuit (to generate local models, or in the initial population of the optimization), the design for the closest number of variables is used, discarding extra variables. If the number of samples is larger than the ones in the design, the remaining samples are drawn randomly.

4.3.2 Gradient Model

The main idea behind the gradient model is that even with an extremely simple model it is possible to improve the optimization kernel by embedding design knowledge into the optimization loop. The gradient model represents the relations between the outputs and the input to which it is more sensitive, i.e., the input with larger effect towards a specific output, and also, the direction of that effect, hence the name, gradient model [19]. The next paragraphs describe the two processing steps used to automatically derive the gradient rules: extracting and ranking the contributions of each input to each output, and, extracting the gradient rules.

4.3.2.1 Extracting and Ranking Design Variables Contributions

In the first step the analysis of the experiments in order to understand which inputs affect most the outputs, henceforward called the main effect, is performed. The main effect is the effect of one independent (input) variable on the dependent (output) variable, when ignoring the effects of all other independent variables [14], where $m_{i,j}$, the main effect of input variable i in the output variable j is computed according to Eq. (4.11):

$$m_{i,j} = \sum_{k=1}^{B^{n-p}} w_{i,k} \times y_k, \quad w_{i,k} = \begin{cases} +1 \, when \, x_{i,k} \geq \frac{B}{2} \\ -1 \, when \, x_{i,k} < \frac{B}{2} \end{cases} \tag{4.11}$$

where k identifies the sample and y the output measure for the sample. When the total main effect of an input variable is positive/negative, there is an indication that if the value of that input variable is increased/decreased, the value of the output will tend to increase/decrease, respectively.

Table 4.8 shows the main effects of the input variables to the outputs (DC Gain and GBW) for the fully factorial DOE matrix in Table 4.5. Table 4.9 shows the main effects obtained from the fractional factorial DOE matrix in Table 4.6. The analysis of the Tables 4.8 and 4.9 shows all variables having a similar and relatively small effect in the gain, while IBias shows a good effect in GBW. Due to their highest absolute value, W_{12} is the input variable that gives more certainty in its effect towards the gain, and IBias the one that gives more certainty in its effect towards the GBW.

4.3.2.2 Extracting the Gradient Rules

Once the main effects are computed, N input variables with the larger contribution to each output are identified as the ones with the large absolute main effect, and then, a refinement procedure is executed. For each output variable y_j, a new DOE

Table 4.8 Main-effect obtained from the full factorial design

Variables	Effect on DC GAIN	$m_{i,1}$
x_1 – W_{12}(m)	−30.61 + 29.72 −30.54 + 29.68 −30.88 + 30.55 −30.8 + 30.48	−2.4
x_2 – W_{34}(m)	−30.61 −29.72 + 30.54 + 29.68 −30.88 −30.55 + 30.8 + 30.48	−0.26
x_3 – IBias (A)	−30.61 −29.72 −30.54 −29.68 + 30.88 + 30.55 + 30.8 + 30.48	2.16
Variables	Effect on GBW	$m_{i,2}$
x_1 – W_{12}(m)	−13.17 + 12 −10.14 + 9.46 −29.89 + 27.51 −23.16 + 21.62	−5.77
x_2 – W_{34}(m)	−13.17 −12 + 10.14 + 9.46 −29.89 −27.51 + 23.16 + 21.62	−18.19
x_3 – IBias (A)	−13.17 −12 −10.14−9.46 + 29.89 + 27.51 + 23.16 + 21.62	57.41

Table 4.9 Main-effect obtained from the fractional factorial designs

Variables	Effect on DC GAIN	$m_{i,1}$
x_1 – W_{12}(m)	−30.61 + 29.68 + 30.55 −30.8	−1.18
x_2 – W_{34}(m)	−30.61 + 29.68 −30.55 + 30.8	−0.68
x_3 – IBias (A)	−30.61 −29.68 + 30.55 + 30.8	1.06
Variables	Effect on GBW	$m_{i,2}$
x_1 – W_{12}(m)	−13.17 + 9.46 + 27.51−23.16	0.64
x_2 – W_{34}(m)	−13.17 + 9.46 −27.51 + 23.16	−8.06
x_3 – IBias (A)	−13.17 −9.46 + 27.51 + 23.16	28.04

matrix is constructed using the fractional factorial sampling, with the N input variables that have the larger contributions as the only elementary variables.

The refinement DOE matrix is then converted to the set of gradient rules for that output variable. This is done by discarding the columns referring to non-elementary variables and transforming the levels of the elementary variables x_i into input gradient symbols $Si_{i,j,k}$ according to:

$$Si_{i,j,k} = \begin{cases} (+)\, when\, x_{i,k} \geq \frac{B}{2} \\ (-)\, when\, x_{i,k} < \frac{B}{2} \end{cases} \tag{4.12}$$

where k identifies the line of the matrix. The output gradient symbols So are converted from the output values as:

$$So_{j,k} = \begin{cases} (+)\, when\, y_{j,k} \geq Y_j^{\text{Max}} - \Delta_j \\ (u)\, when \left(Y_j^{\text{Min}} + \Delta_j\right) < y_{j,k} < \left(Y_j^{\text{Max}} - \Delta_j\right) \\ (-)\, when\, y_{j,k} \leq Y_j^{\text{Min}} + \Delta_j \end{cases} \tag{4.13}$$

where Y_j^{Max} and Y_j^{Min} are respectively the maximum and minimum values of the output y_j obtained in the DOE matrix (not the refinement matrix), and Δj is $|Y_j^{\text{Max}} - Y_j^{\text{Min}}|/4$. The meaning of the symbols is: (−) a decrease; (+) increase and (U) undefined. Table 4.10 illustrates the Eq. (4.13) using Table 4.6 as matrix of refinement, while Table 4.11 illustrates the Eq. (4.13) using Table 4.7 as matrix of refinement.

Table 4.10 Extraction of gradient rules for DC Gain

$y_{j,k}$	$Y_j^{\text{Max}} - \Delta_j$	$Y_j^{\text{Min}} + \Delta_j$	$So_{j,k}$
$y_{1,1}$ = 30.61	30.58	29.98	$So_{1,1}$: (+)
$y_{1,2}$ = 29.68			$So_{1,2}$: (−)
$y_{1,3}$ = 30.80			$So_{1,3}$: (+)
$y_{1,4}$ = 30.55			$So_{1,4}$: (U)

Table 4.11 Extraction of gradient rules for GBW

$y_{j,k}$	$Y_j^{\text{Max}} - \Delta_j$	$Y_j^{\text{Min}} + \Delta_j$	$So_{j,k}$
$y_{2,1}$ = 13.17	24.78	14.57	$So_{2,1}$: (−)
$y_{2,2}$ = 9.46			$So_{2,2}$: (−)
$y_{2,3}$ = 27.51			$So_{2,3}$: (+)
$y_{2,4}$ = 23.16			$So_{2,4}$: (U)

Table 4.12 Set of gradient rules for DC Gain

	$Si_{1,1}$ W_{12}	$Si_{2,1}$ IBias	So_1 DC Gain
1	(−)	(−)	(+)
2	(+)	(−)	(−)
3	(−)	(+)	(+)
4	(+)	(+)	(U)

Table 4.13 Set of gradient rules for GBW

	$Si_{1,2}$ W_{34}	$Si_{2,2}$ IBias	So_2 GBW
1	(−)	(−)	(−)
2	(+)	(−)	(−)
3	(−)	(+)	(+)
4	(+)	(+)	(U)

Tables 4.12 and 4.13 illustrates the corresponding gradient rules. In both cases rule 4 is useless because it have undefined meaning. For the DC Gain rules, 1 and 3 make sense, rule 2 is also useless as it is never an objective to decrease the DC Gain. With respect to GBW, only rule 3 makes sense in driving it up.

4.3.2.3 Combining the Gradient Rules with the Optimization Kernel

The developed approach combines the gradient model with the multi-objective multi-constraint optimization kernel. The implementation is made through the use of a new mutation/neighbour/turbulence operator, which takes into account the gradient model to perform changes to the elements selected to be mutated. The chromosome, i.e., the collection of optimization or design variables, is represented by a vector of continuous variables $[x_1, \ldots, x_n]$. The classical operation corresponds to random changes applied to a selection of those continuous variables. Here, the gradient model is used to introduce design knowledge into the operator to speed up the convergence of the algorithm.

The reference operator implemented in AIDA-C is based in the continuous-valued mutation operator introduced Deb and Agrawal in [20]. In this operator δ_i, the perturbation applied, is defined as $\delta_i = (x_i^M - x_i)/(X_i^{Max} - X_i^{Min})$, where x_i^M and x_i are the perturbed and original values, respectively. Moreover, δ_i is a random variable, with values in the interval of −1 to 1, and p.d.f.:

$$P(\delta) = 0.5 \times (\eta + 1) \times (1 - |\delta|)^{\eta} \tag{4.14}$$

where η is a parameter used to control the spread of the distribution, Fig. 4.7 shows the p.d.f. for various values of η.

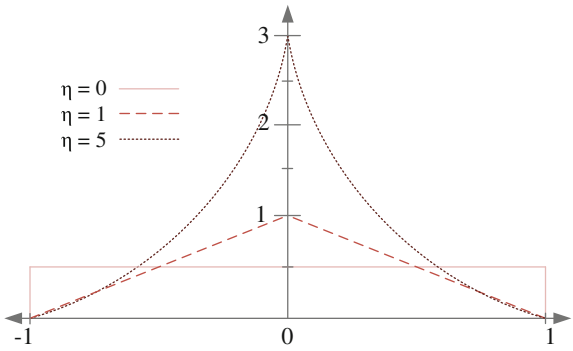


Fig. 4.7 Perturbation p.d.f

A realization of δ, $\overline{\delta}$ is obtained from a uniform random number $u \in [0,\ 1[$ using Eq. (4.15), which is obtained from Eq. (4.14) by solving $\int_{-1}^{\overline{\delta}} P(\delta) = u$.

$$\overline{\delta} = \begin{cases} (2u)^{\frac{2}{\eta+2}} - 1, & if\ u < 0.5 \\ 1 - [2(1-u)]^{\frac{2}{\eta+2}}, & if\ u \geq 0.5 \end{cases} \tag{4.15}$$

leading to the perturbed value, x_i^M, is given by $x_i^M = x_i + \overline{\delta}\left(X_i^{\text{Max}} - X_i^{\text{Min}}\right)$. The gradient rules are then applied. The application of the rules follows the expression in Eq. (4.16):

$$x_i^G = (1 + \mu.\gamma(Si_i))\, x_i^M \tag{4.16}$$

where x_i^G is the variable value after the application of the rule, $\gamma(Si_i)$ is a function of the gradient symbol defined in Eq. (4.17), and $u \in [0, c[$ is a uniformly distributed random number between 0 and c, the change rate parameter.

$$\gamma(Si_i) = \begin{cases} +1\ when\ Si_i = (+) \\ -1\ when\ Si_i = (-) \end{cases} \tag{4.17}$$

The application of the gradient rules is dependent on the existence of a suitable rule for the optimization target, i.e., it is irrelevant to have a rule to decrease the gain, if the target is to increase it. The rules are selected by searching, for each optimization objective, if there is a rule that causes the desired effect in the corresponding response variable. If this rule is found, then the variables with larger contributions are affected as described before. Figure 4.8 illustrates an example application of a gradient rule in the mutation operator.

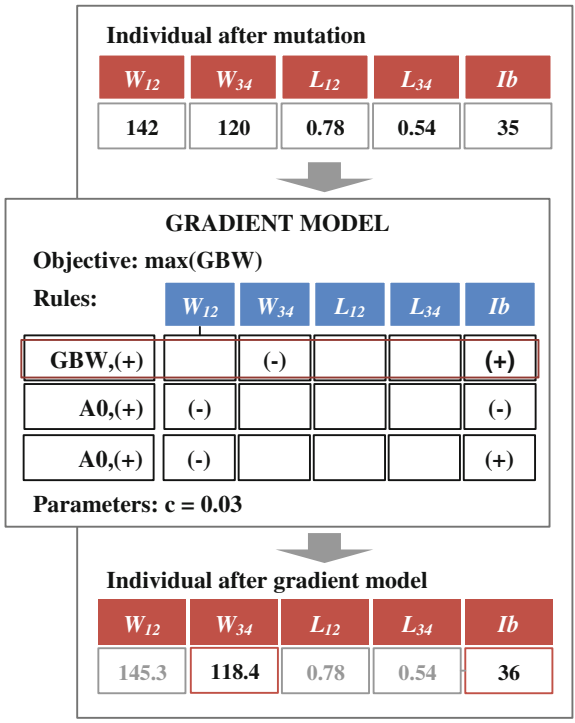


Fig. 4.8 Gradient rules in the mutation operator

4.4 Conclusions

The multi-objective nature of the IC design sizing makes it well suited for automatic design using multi-objective optimization strategies. In this chapter the multi-objective optimization kernels, NSGA-II, MOPSO, MOSA and the kernel hybridization implemented in AIDA-C were described, taking into consideration how the circuit sizing is handled as an optimization problem, even when considering the extra constraints introduced by PTV corners. Finally, this chapter describes how the optimization process is enhanced with the machine learning techniques that add automatically generated design knowledge to guide the optimizer towards better solutions faster.

References

1. Lourenço N, Horta N (2012) GENOM-POF: multi-objective evolutionary synthesis of analog ICs with corners validation. In: Genetic and evolutionary computation conference, Philadelphia, 2012
2. Debyser G, Gielen G (1998) Efficient analog circuit synthesis with simultaneous yield and robustness optimization. In: 1998 IEEE/ACM international conference on computer-aided design (ICCAD 98), San Jose, CA, 8–12 Nov 1998

3. Dharchoudhury A, Kang SM (1995) Worst-case analysis and optimization of VLSI circuit performances. IEEE Trans Comput Aided Des Integr Circuits Syst 14(4):481–492
4. Antreich KJ, Graeb HE, Wieser CU (1994) Circuit analysis and optimization driven by worst-case distances. IEEE Trans Comput Aided Des Integr Circuits Syst 13(1):57–71
5. Schwencker R, Schenkel F, Pronath M, Graeb H (2002) Analog circuit sizing using adaptive worst-case parameter sets. In: Design, automation and test in Europe conference and exhibition, Paris, 4–8 March 2002
6. Graeb HE (2007) Analog design centering and sizing. Springer, Netherlands
7. McConaghy T, Breen K, Dyck J, Gupta A (2013) Variation-aware design of custom integrated circuits: a hands-on field guide. Springer, New York
8. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
9. Reyes-Sierra M, Coello Coello CA (2006) Multi-objective particle swarm optimizers: a survey of the state-of-the-art. Int J Comput Intell Res 2(3):287–308
10. Bandyopadhyay S, Saha S (2013) Some single- and multi-objective optimization techniques. Unsupervised classification. Springer, Berlin, Heidelberg, pp 17–58
11. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
12. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equation of state calculations by fast computing machines. J Chem Phys 21(6):1087–1092
13. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of the 1995 IEEE international conference on neural networks, Perth, 1995
14. Montgomery DC (2001) Design and analysis of experiments, 5th edn. John Wiley and Sons, New York
15. Hernandez AS (2008) Breaking barriers to design dimensions in nearly orthogonal latin hypercubes. Dissertation, Naval Postgraduate School, Monterey CA
16. Cioppa TM, Lucas TW (2207) Efficient nearly orthogonal and space-filling latin hypercubes. Technometrics 49(1):45–55
17. Min-Qian Liu Jinyu Yang (2012) Construction of orthogonal and nearly orthogonal Latin hypercube designs from orthogonal designs. Statistica Sinica 22(1):433–442
18. Nguyen Nam-Ky, Lin Dennis K J (2012) A note on near-orthogonal latin hypercubes with good space-filling properties. J Stat Theory Pract 6(3):492–500
19. Rocha F, Lourenço N, Póvoa R, Martins R, Horta N (2014) A new metaheuristc combining gradient models with NSGA-II to enhance analog IC synthesis. In: 2013 IEEE Congress on evolutionary computation, Cancun, 2013
20. Deb K, Agrawal RB (1995) Simulated binary crossover for continuous search space. Complex Syst 9(2):115–148

Chapter 5
AIDA-C Circuit Sizing Results

5.1 Evolutionary Parameters Impact

Though AIDA-C supports several optimization kernels, the most relevant is NSGA-II, as will become clearer in Sect. 5.4. In this section the results obtained from testing the impact of the evolutionary engine parameters on the synthesis of analog circuits are introduced. This first study focus on parameters of the optimization kernel: population size, number of generations and, mutation and crossover rates.

5.1.1 Crossover and Mutation Rates

A single ended folded-cascode amplifier for the United Microelectronics Corporation (UMC) 180 nm process was used to conduct parametric sweeps over the mutation and crossover probability. The circuit and test-bench schematics are shown in Fig. 5.1, and the ranges, objectives and constraints are listed in Tables 5.1 and 5.2.

The circuit setup specifies an optimization problem with 15 optimization variables that define the combinatorial search space represented in (5.1), which counts all the possible combinations of variables' values within the defined ranges.

$$\begin{aligned} &\{0.18, 0.185, \ldots, 5\}^6 \times \{0.24, 0.48, \ldots, 200\}^6 \\ &\times \{30, 40, \ldots 400\} \times \{-0.4, -0.39, \ldots, 0\} \times \{0, 0.01, \ldots, 0.40\} \\ &= 98^6 \times 834^6 \times 38 \times 41 \times 41 = 1.9 \times 10^{34} \end{aligned} \tag{5.1}$$

The multi-objective multi-constraint optimization problem derived from the circuit specifications using the method described in Chap. 4 of this book is illustrated in (5.2) and (5.3), showing how the conversion is made. As there is a direct correspondence between the optimization problem and the sizing specifications in the remaining of this chapter only the circuit specifications are shown. In this case

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0_5

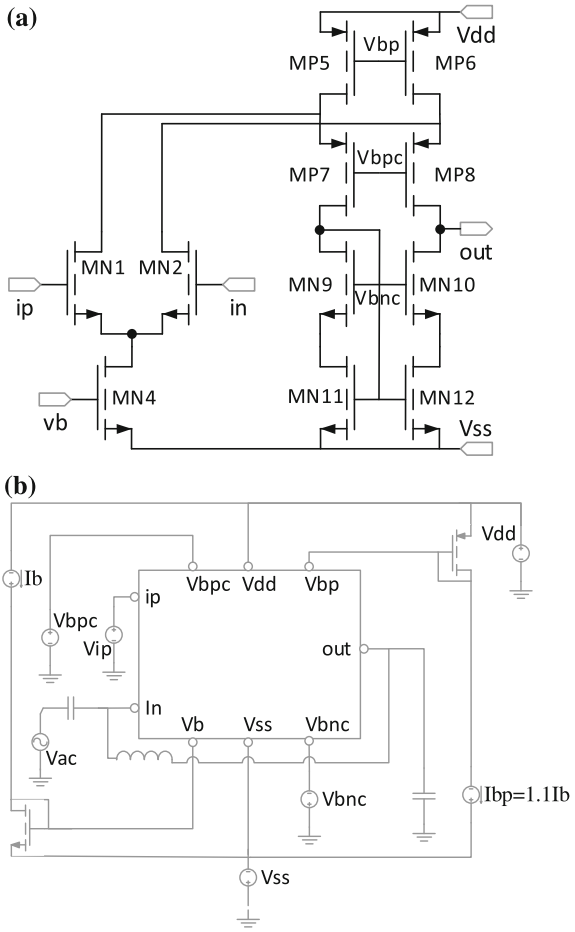


Fig. 5.1 Single-ended folded cascade amplifier; **a** schematic; **b** test-bench

Table 5.1 Single-ended folded cascode amplifier variable ranges

Variable (Unit)	Min.	Grid resolution	Max.
l1, l4, l5, l7, l9, l11 (nm)	180	50	5,000
w1 w4 w5 w7 w9 w11 (μm)	0.24	0.24	200
Ib (μA)	30	10	400
Vbcn (V)	−0.4	0.01	0.0
Vbcp (V)	0.0	0.01	0.4

The variables l1 and w1 are dimensions of MN1 and MN2; l4 and w4 of MN4; l5 and w5 of MP5 and MP6; l7 and w7 of MP7 and MP8; l9 and w9 of MN9 and MN10; l11 and w11 of MN11 and MN12

Table 5.2 Single-ended folded cascode amplifier specifications

Specifications	Measure	Target	Units	Description
Objectives	RMS(Power)	Minimize	mW	Power
	Sqrt(Area)	Minimize	µm	Unity-gain frequency
Constraints	GDC	≥ 70	dB	DC gain
	GBW	≥ 12	MHz	Unity-gain frequency
	PM	$55 \leq$ PM ≤ 90	°	Phase margin
	SR	≥ 10	V/µs	Slew Rate
	vov[1]	≥ 30	mV	Vgs − Vt
	d[1]	≥ 1.2	mV	(Vds − Vdsat)/Vdsat
	Osp	≥ 0.5	V	Vbcp + $\lvert Vt^{MP8}\rvert$
	Osn	≤ -0.5	V	Vbcn − $\lvert Vt^{MN10}\rvert$

[1]The constraint applies to: MN1, MN4, MP5, MP7, MN9 and MN11

the design specifications lead to the optimization problem with 2 objectives and 19 constraints.

$$\begin{aligned} f_1(x) &= sqrtarea \\ f_2(x) &= rmspower \end{aligned} \tag{5.2}$$

$$\begin{aligned} g_1(x) &= \frac{GDC - 70dB}{|70dB|}, g_2(x) = \frac{GBW - 12MHz}{|12MHz|} \\ g_3(x) &= \frac{PM - 55^\circ}{|55^\circ|}, g_4(x) = \frac{90^\circ - PM}{|90^\circ|}, g_5(x) = \frac{SR - 10V/\mu s}{|10V/\mu s|} \\ g_6(x) &= \frac{VOV_{MN1} - 30mV}{|30mV|}, g_7(x) = \frac{DP_{MN1} - 1.2}{|1.2|}, g_8(x) = \frac{VOV_{MN4} - 30mV}{|30mV|} \\ , g_9(x) &= \frac{DP_{MN4} - 1.2}{|1.2|}, g_{10}(x) = \frac{VOV_{MP5} - 30mV}{|30mV|}, g_{11}(x) = \frac{DP_{MP5} - 1.2}{|1.2|} \\ g_{12}(x) &= \frac{VOV_{MP7} - 30mV}{|30mV|}, g_{13}(x) = \frac{DP_{MP7} - 1.2}{|1.2|}, g_{14}(x) = \frac{VOV_{MN9} - 30mV}{|30mV|} \\ g_{15}(x) &= \frac{DP_{MN9} - 1.2}{|1.2|}, g_{16}(x) = \frac{VOV_{MN11} - 30mV}{|30mV|}, g_{17}(x) = \frac{DP_{MN11} - 1.2}{|1.2|} \\ g_{18}(x) &= \frac{OSP - 0.5V}{|0.5V|}, g_{19}(x) = \frac{-0.5V - OSN}{|-0.5V|} \end{aligned} \tag{5.3}$$

Multiple runs, varying the values for the evolutionary parameters were executed to study the impact and tune the parameter. Figures 5.2 and 5.3 show the effects of varying the crossover and mutation rates, respectively.

The % of crossover did not affect the Pareto optimal front (POF) significantly for values larger than 50 %, nevertheless larger crossovers rates (>90 %) yield better results. This result fit the general experience with evolutionary algorithms where the crossover is usually high. The mutation rate around 25 % (green and orange POFs)

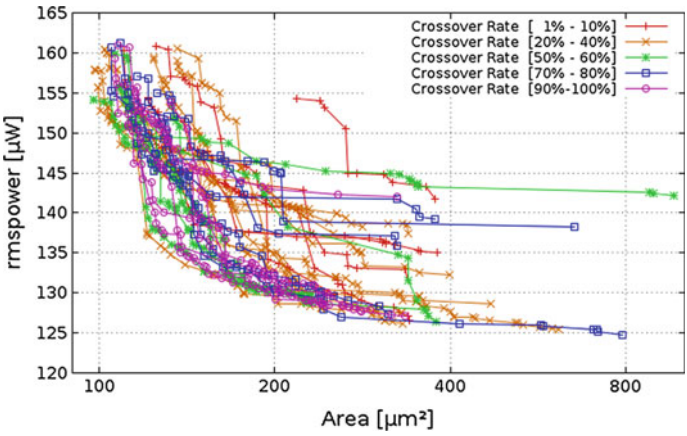


Fig. 5.2 POFs for various crossover rates (32 elem, 200 gen)

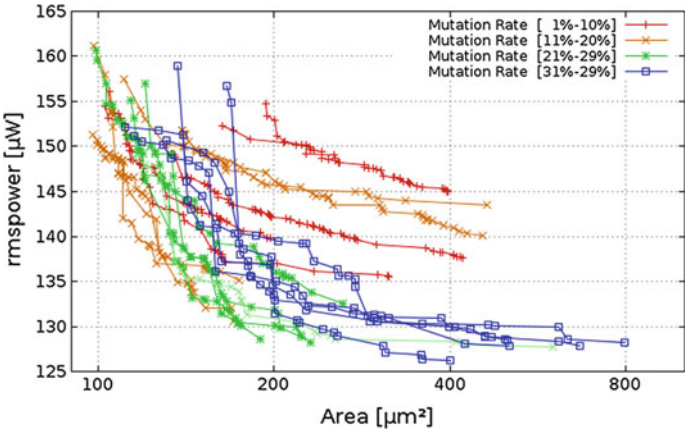


Fig. 5.3 POFs for various mutation % (32 elem, 200 gen)

presented better results, at first this value seems high, however the mutation operator is not just the generation of a random value within the specified range. In simulated binary crossover (SBX) the mutation generates a value around the original value with a bell shape probability distribution. A detailed description of the mutation operator is found in Chap. 4 of this book together with the Gradient model. In all examples in this Chapter and in Chap. 7 of this book, if not mentioned otherwise, the % of crossover is set 90 % and the mutation rate to 30 %. Throughout the diverse executions of the tool, there was no further need to change either of these parameters.

5.1.2 Population Size and Number of Generations

To evaluate the dispersion of the solutions due to the population size (P) and number of generations (G), the single stage amplifier with gain enhancement using voltage combiners (VCs) proposed in [1] was optimized to maximize the figure of merit (FOM) and maximize de low-frequency gain (GDC). The circuit schematic is shown in Fig. 5.4 and the performance figures are measured using Mentor Graphics Eldo® for the UMC 130 nm process. The design variable and ranges are shown in Table 5.3, and the specifications are presented in Table 5.4.

By maximizing the FOM, the power consumption is minimized as well as the gain-bandwidth product (GBW) is maximized, as described by the FOM (5.4), where C_{load} is the load capacity and IDD is the current consumption.

$$FOM = \frac{GBW \times C_{load}}{IDD} \left[\frac{MHz \times pF}{mA}\right] \tag{5.4}$$

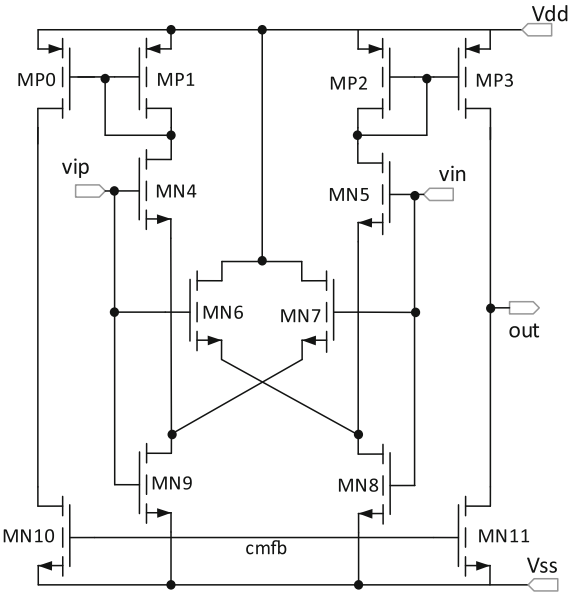


Fig. 5.4 Single stage amplifier with gain enhancement using voltage combiners

Table 5.3 Single stage amplifier with gain enhancement using VCs: variable and ranges

Variable (Unit)	Min.	Grid resolution	Max.
l0, l1, l4, l6, l8, l10 (nm)	120	10	1000
w0 w1 w4 w6 w8 w10 (μm)	1	0.1	10
nf0, nf1, nf4, nf6, nf8, nf10	1	2	8

The variables l0, w0, and nf0 are dimensions of MP0 and MP3; l1, w1, and nf1 of MP1 and MP2; l4, w4 and nf4 of MN4 and MN5; l6, w6 and nf6 of MN6 and MN7; l8, w8, and nf8 of MN8 and MN9; l10 w10 and nf10 of MN10 and MN11

Table 5.4 Single stage amplifier with gain enhancement using VCs: specifications

Specifications	Measure	Target	Units	Description
Objectives	GDC	Maximize	mW	DC gain
	FOM	Maximize	MHz · pF/mA	Figure of merit
Constraints	FOM	≥ 1000	MHz · pF/mA	Figure of merit
	IDD	≤ 350	μA	Current consumption
	GDCc	≥ 40	dB	DC gain
	GBW@6pF	≥ 30	MHz	Unity-gain frequency
	PM	≥ 60	°	Phase margin
	OVP[1]	≥ 100	mV	Overdrive voltages of the PMOS devices ($V_{TH} - V_{GS}$)
	OVN[2]	≥ 100	mV	Overdrive voltages of the NMOS Devices ($V_{GS} - V_{th}$)
	DP[1]	≥ 50	mV	Saturation margin of the PMOS devices ($V_{DSat} - V_{DS}$)
	DN[2]	≥ 50	mV	Saturation margin of the NMOS devices ($V_{DS} - V_{DSat}$)

[1]Applies to: MP0, MP1, MP2 and MP3
[2]Applies to: MN4, MN5, MN6, MN7, MN8, MN9, MN10 and MN11

The criteria in setting the constraints were: all devices are set to operate in the moderate/strong inversion region, therefore, their overdrive voltage is required to be greater or equal to 100 mV, while all devices are to be working in saturation hence their saturation margin is required to be greater or equal to 50 mV. To ensure stability the phase margin must be larger than 60°, and a 50 dB gain must be ensured.

These design specifications lead to the optimization problem with 18 optimization variables, 2 objectives (GDC and FOM) and 29 constraints shown in (5.5) and (5.6).

$$\begin{aligned} f_1(x) &= -GDC \\ f_2(x) &= -FOM \end{aligned} \tag{5.5}$$

$$\begin{aligned} g_1(x) &= \frac{FOM - 1000\ MHz.pF/mA}{|-\ 1000\ MHz.pF/mA|} \\ g_2(x) &= \frac{350\mu A - IDD}{|350\mu A|}, g_3(x) = \frac{GDC - 50dB}{|50dB|} \\ g_4(x) &= \frac{GBW - 30MHz}{|30MHz|}, g_5(x) = \frac{PM - 60^\circ}{|60^\circ|} \\ g_{6+d}(x) &= \frac{OVP^d - 100mV}{|100mV|}, g_{10+d}(x) = \frac{DP^d - 50mV}{|50mV|} d = 0:3 \\ g_{10+d}(x) &= \frac{OVN^d - 100mV}{|100mV|}, g_{22+d}(x) = \frac{DN^d - 50mV}{|50mV|} d = 4:11 \end{aligned} \tag{5.6}$$

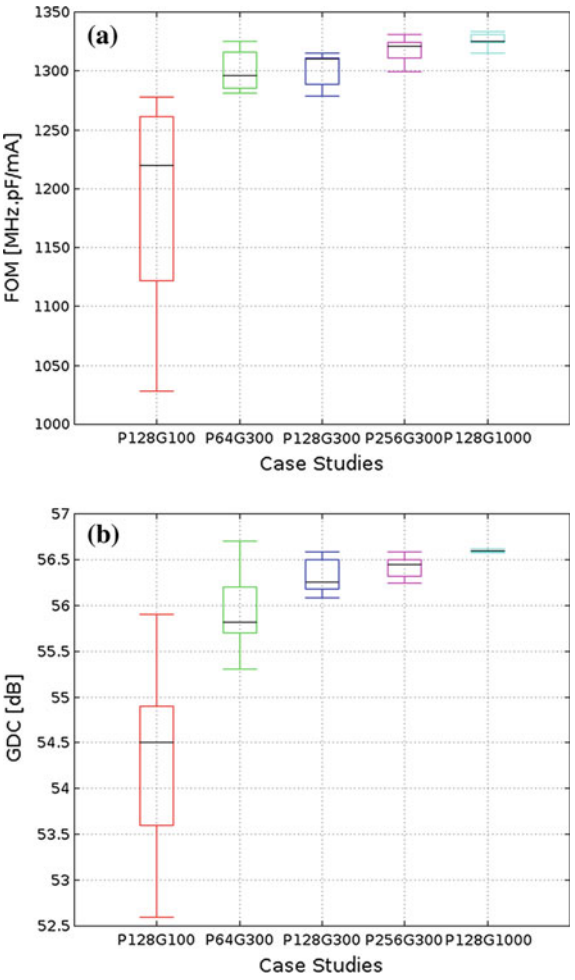


Fig. 5.5 Dispersion of the POF limits: **a** Maximum FOM for various P/G setups; **b** Maximum GDC for various P/G setups

Ten runs were done for 5 different setups of population size and number of generation. The pairs tested were P128G100, P64G300, P128G300, P256G300 and P128G1000. Figure 5.5a compares the best value of the FOM objective obtained with each setup, and Fig. 5.5b shows the same, but for the GDC objective. Figure 5.6 shows the 10 Pareto fronts obtained for the setup P128G1000.

In terms of population size and number of generations, the progression of the best values with the total number of evaluations (P*G) shows a logarithmic behavior as can be seen in Fig. 5.5. In the P128G100 setup there is a great variability between runs. The cause of this variability is the small number of generation, as setups with the same population size (P128G300 and P128G1000) show multiple runs of the stochastic optimization resulting in approximately the same solutions.

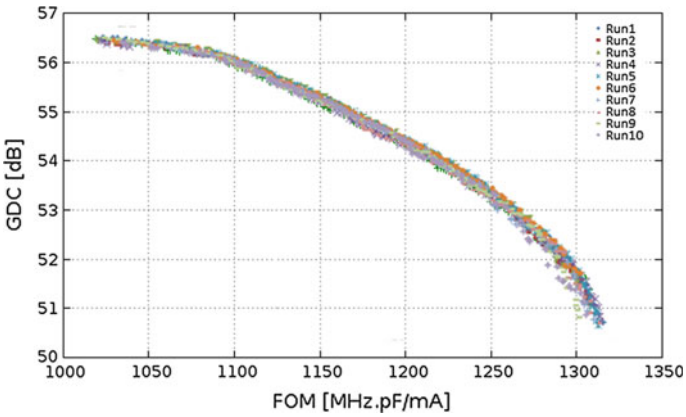


Fig. 5.6 10 POF obtained with P = 128 and G = 1000

It is interesting to note that the setup P64G300 performs comparably with the setup P128G300 despite requiring half the evaluations (which means half the execution time). Analyzing the remaining of the plot shows that increasing the number of evaluations increased both the robustness and quality of the solution (obviously at expense of the number of simulations).

The populations size and number of generations, and respective total number of simulations, are trickier to derive. With the smallest population that is large enough not to converge prematurely, the required number of generations is reduced, however, there is more variability between runs and the risk for premature convergence. On the other side, larger populations avoid premature convergence but more generations are needed spread the "good" genes through the population, making the overall execution time slower in twofold: more simulations per generation and more generations.

5.2 Comparing the Evaluation Strategies

So far, the tests were conducted considering nominal conditions only. In this section the impact of considering the worst case parameter sets in the evaluation of circuit performance in terms of the solution's performance and execution time is explored. This study focus on the differences between the optimization flows Typical (T), Corners (C) and Typical plus Corners (TC). In T the optimization uses only the nominal conditions; in C the optimization considers the corner cases; and in TC two optimization jobs are executed sequentially, first the optimization considers only nominal conditions, then, the results are used as a starting point for the corner optimization.

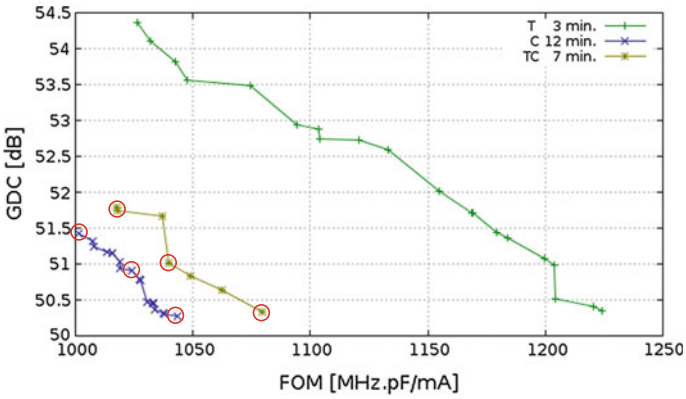


Fig. 5.7 POF obtained using the 3 design strategies T, TC and C

5.2.1 Amplifier with Gain Enhancement Using VCs

The Single-Stage Amplifier with Gain Enhancement using VCs introduced in Sect. 5.1.2 of this Chapter is here used for nominal and worst case sizing. The specifications are the same as the ones in Table 5.4. In addition, the following four corner conditions are considered for devices' models: slow NMOS and slow PMOS (SS); slow NMOS and fast PMOS (SNFP); fast NMOS and slow PMOS (FNSP); and, fast NMOS and fast PMOS (FF). Figure 5.7 illustrates the Pareto fronts and execution time that were obtained from one execution of each strategy: T, C and TC. The algorithm parameters were: population of 32 elements and 400 generations for T and C, and 200/200 for TC (200 for the first step and 200 for the second step) for a fast execution.

The nominal POF obtained for the amplifier using voltage combiner was found faster (in approx. 3 min), and dominates the others (because it has fewer constraints) and does note account not the variability of the process. TC strategy was faster than C and resulted in better solutions. By starting the corner optimization from the already optimized typical POF, it is easier to fulfill the additional constraints imposed by the corners and leads to better and faster results, however there is some biasing that narrows the search range. The number of feasible solutions is much smaller in TC (7) than in either T (18) or C (19). Table 5.5 summarizes the output of C and TC strategies.

5.2.2 Fully Differential Telescopic Amplifier

A fully differential telescopic amplifier circuit including bias for the UMC 180 nm technology was also considered in this analysis. The circuit's schematic is shown in

Table 5.5 Summary of the corner and typical plus corner run illustrated in Fig. 5.7

Strategy	Corner(C)			Typical plus corner (TC)		
	Small FOM	Middle	High FOM	Small FOM	Middle	High FOM
Time (s)	12			7		
FOM (MHZ · pf/mA)	1001.5	1024.2	1043.4	1017.7	1039.9	1079.5
Gain (dB)	51.43	50.91	50.28	51.78	51.01	50.33
L0 (μm)	0.64	0.60	0.56	0.63	0.57	0.53
W0 (μm)	66.4	64.4	62.4	42.3	45.0	45.0
NF0	8	8	8	8	6	6
L1 (μm)	0.37	0.37	0.36	0.30	0.30	0.31
W1 (μm)	6.3	6.3	6.3	3.3	3.6	3.6
NF1	4	4	4	4	2	2
L4 (μm)	0.49	0.50	0.50	0.70	0.69	0.72
W4 (μm)	9.8	10.0	10.0	16.1	15.5	14.0
NF4	2	4	2	4	4	4
L6 (μm)	0.50	0.49	0.49	0.71	0.73	0.73
W6 (μm)	37.1	37.7	37.3	53.1	53.3	53.6
NF6	8	8	8	2	2	2
L8 (μm)	0.94	0.94	0.94	0.94	0.94	0.94
W8 (μm)	1.0	1.0	1.0	1.0	1.0	1.0
NF8	2	2	2	2	2	2
L10 (μm)	0.94	0.94	0.91	0.93	0.88	0.93
W10 (μm)	5.4	4.4	4.6	10.1	9.9	2.7
NF10	2	2	2	4	4	4

Fig. 5.8, and the variable and ranges, and specifications are listed in Tables 5.6 and 5.7. The problem has 16 optimization variables, 3 objectives and 38 constraints. 8 corner cases were defined using the extreme combination of technology models (typical, fast and slow), voltage supply (1.7, 1.75, 1.8 V) and temperature values (−40, 50, 120 °C).

In Fig. 5.9, the projections of Gain versus Power and GBW versus Power are overlapped, each point in the Pareto front is represented by the 2 points in the graphic that have the same value of Power, one from each of the projections. By having more than 2 objectives the 2-D projections are not monotonic like in the previous example. This is due to the fact that there are solutions that seem to be dominated but have better performance in the objectives not present in that projection. The objective POF was obtained using the TC strategy with parameters: population of 80 elements and 300/200 generations, and shows how the effect of corners cases in decreasing the optimum performance that can be achieved by the circuit. Table 5.8 summarizes the sizing results.

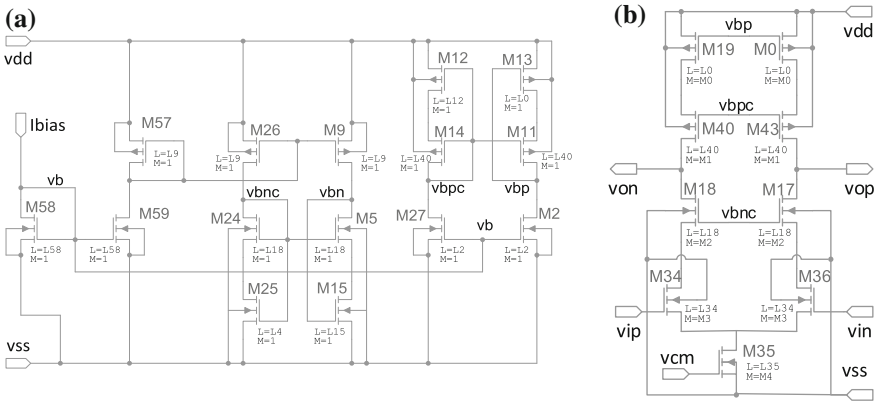


Fig. 5.8 Fully differential telescopic amplifier schematic: **a** bias; **b** amplifier

Table 5.6 Fully differential telescopic amplifier: variable's ranges

Variable (Unit)	Min.	Grid resolution	Max.
l0, l2, l9, l12, l15, l18, l24, l34, l35, l40, l58 (μm)	0.180	0.01	10
m0, m1, m2, m3, m4	1	2	200

All devices have a fixed width of 2 μm

Table 5.7 Fully differential telescopic amplifier: specifications

Specifications	Measure	Target	Units	Description
Objectives	Power	Minimize	W	Power
	Gbw	Maximize	Hz	Unity-gain frequency
	gain_dc	Maximize	dB	DC gain
Constraints	gain_dc	≥ 75	dB	DC gain
	Gbw	≥ 100	MHz	Unity-gain frequency
	Fase	$60 \leq$ fase ≤ 90	°	Phase margin
	Power	≤ 10	mW	Power
	Iavdd	≤ 10	mA	Current consumption
	vov[1]	≥ 100	mV	Vgs − Vt
	vov_m18, vov_m17	≥ 45	mV	Vgs − Vt
	vov_m34, vov_m36	≥ 50	mV	Vgs − Vt
	vov[2]	≤ 200	mV	Vgs − Vt
	vov[3]	≤ 300	mV	Vgs − Vt
	d[4]	$50 \leq d \leq 200$	mV	Vds − Vdsat
	d[5]	≥ 50	mV	Vds − Vdsat

[1]The constraint applies to: M19, M0, M40, M43 and M35
[2]The constraint applies to: M19, M0, M18 and M17
[3]The constraint applies to: M40, M43, M34, M36 and M35
[4]The constraint applies to: M40, M43, M17, M18 and M35
[5]The constraint applies to: M19, M0, M34, and M36

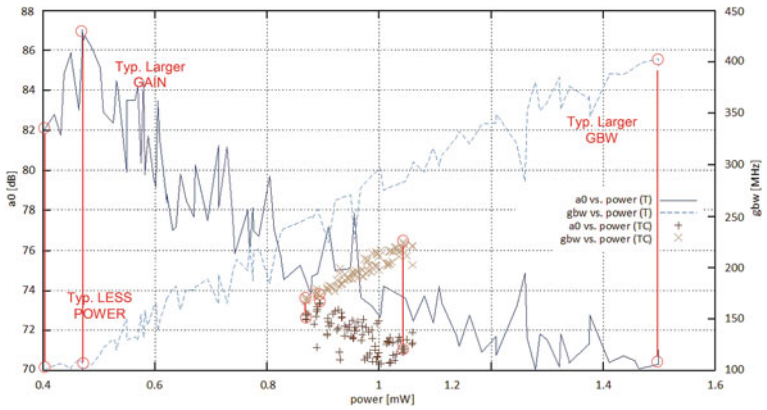


Fig. 5.9 2D Projections of the 3D POF obtained using T and TC

Table 5.8 Summary of the synthesis results

Strategy	Typical step [637 (s)]			Corner step [5363 (s)]		
	Less power	Larger GBW	Larger gain	Less power	Larger GBW	Larger gain
Power (mW)	0.402	1.498	0.470	0.869	1.04	0.894
GBW (MHz)	100.8	394.8	100.8	170.1	224.6	172.0
Gain (dB)	81.98	71.02	87.02	72.53	70.87	73.32

The consideration of worst case parameter sets implies more simulations for the same number evaluations making the overall sizing slower. The usage of a pre-optimized nominal front helps to reduce this effect. In both cases several solutions that meet the specification when considering the worst case corners are found, nevertheless the performance degradation is notorious.

5.3 Gradient Model

The gradient model was introduced to improve the efficiency of the optimization kernel. The results obtained with this enhancement are shown in this section.

5.3.1 Folded Cascode Amplifier

The folded cascode amplifier introduced in Sect. 5.1.1 was migrated for UMC 130 nm and was optimized with and without the gradient model for the specification in Table 5.9. In this study, the model was generated with a base of two

Table 5.9 Single-ended folded cascode amplifier: specifications II

Specifications	Measure	Target	Units	Description
Objectives	GDC	Maximize	dB	DC gain
	Area	Minimize	m^2	Area of devices
Constraints	GDCc	≥ 80	dB	DC gain
	GBW	≥ 12	MHz	Unity-gain frequency
	PM	$55 \leq PM \leq 90$	°	Phase margin
	SR	≥ 10	V/μs	Slew Rate
	vov[1]	≥ 30	mV	Vgs − Vt
	d[1]	≥ 1.2	mV	(Vds − Vdsat)/Vdsat
	osp	≥ 0.0	V	Vbcp + $\lvert Vt^{MP8} \rvert$
	osn	≤ -0.3	V	Vbcn − $\lvert Vt^{MN10} \rvert$

[1]The constraint applies to: MN1, MN4, MP5, MP7, MN9 and MN11

Table 5.10 Gradient rules

Target	Variable/Gradient
GDC (−)	L9 (−)
GDC (+)	L9 (+)
area (−)	W11 (−)
area (+)	W11 (+)

(B = 2) and considering only the design variable with larger contribution (N = 1). The extracted gradient rules for the optimization objectives are shown in Table 5.10, the model was generated in less than 5 min. Figure 5.10 illustrates the improvements achieved by the use of the gradient model combined with the mutation operator. Moreover, it shows that with the gradient model AIDA-C achieves better solutions at generation 2,000 than without it at generation 2,000 or 4,000, even for 60,000 generations AIDA-C without the model cannot reach the maximum DC Gain attained with the use of the model.

To confirm that this is not an isolated case, 20 executions with different seeds were done. The results are shown in Fig. 5.11 where it can be seen that the inclusion of the gradient model consistently lead to better solutions. Table 5.11 shows the average over the 20 run of: the number of point in the final POF, and the normalized non-dominated area, which measures the ratio between the non-dominated and dominated area in the performance plane. It confirms the analysis of Fig. 5.11, where the usage of the gradient model leads to more and better solutions.

5.3.2 *Amplifier with Gain Enhancement Using VCs*

The Single-Stage Amplifier with Gain Enhancement using VCs described in Sect. 5.1.2 is here used again, in this case for nominal sizing with and without the

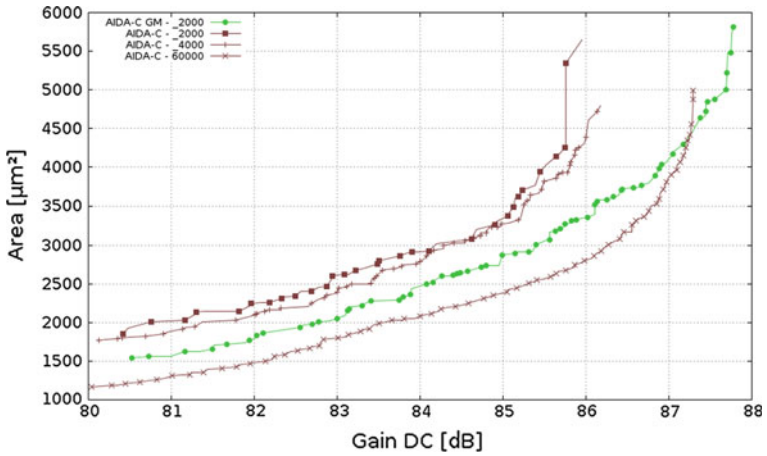


Fig. 5.10 AIDA-C (for 60,000, 4,000 and 2,000 generations) versus AIDA-C GM (for 2,000 generations)

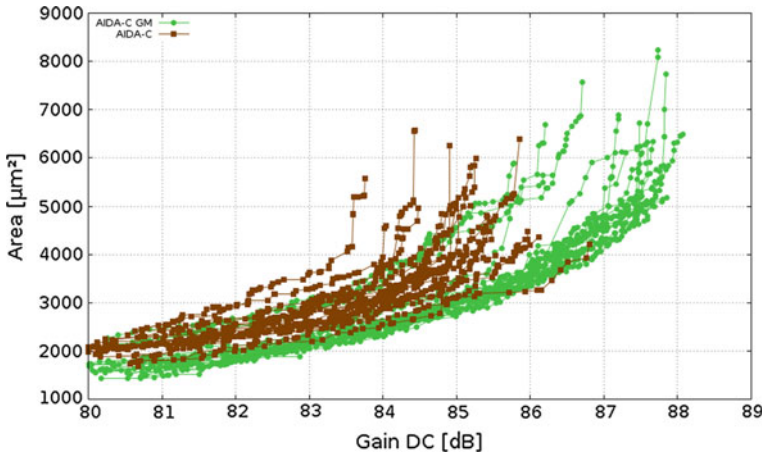


Fig. 5.11 AIDA-C versus AIDA-C + GM for 20 different initial populations (for 2,000 generations)

Table 5.11 AIDA-C versus AIDA-C + GM (2,000 generations)

	Nr. Points POF	Non-dominated area
AIDA-C	51.55	0.426
AIDAC + GM	81.7	0.2

Gradient Model. Again, the specifications are the ones describe in Table 5.4. The algorithm parameters were: population of 128 elements and 1,000 generations. Figure 5.12 shows the Pareto fronts obtained with 10 executions for each case. Again the use of the Gradient Model consistently resulted in improved results.

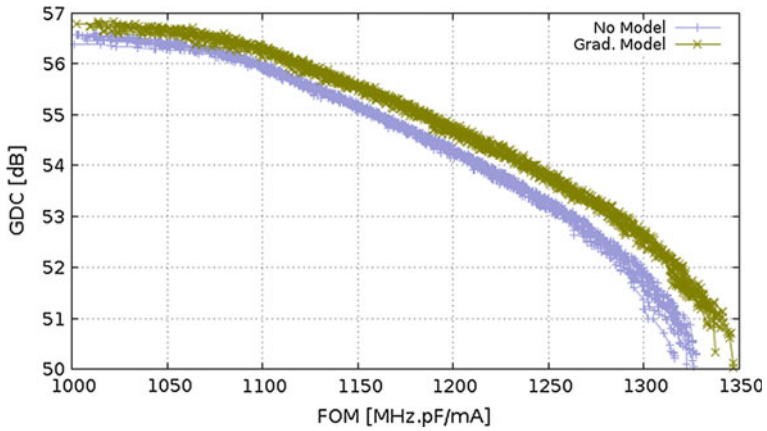


Fig. 5.12 Amplifier with gain enhancement using voltage combiners POF with gradient model

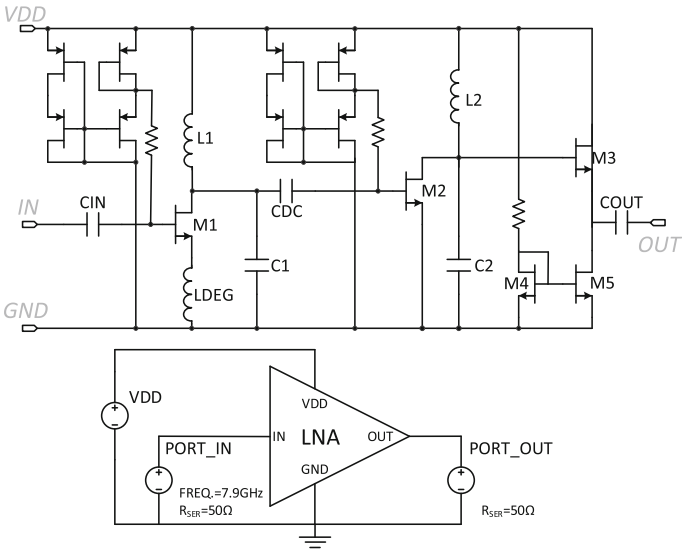


Fig. 5.13 Schematic and test-bench of the 7.9 GHz LNA

5.3.3 *Low Noise Amplifier*

The 1.2 V low noise amplifier operating 7.9 GHz illustrated in Fig. 5.13 is considered in this test case. This case study includes 21 input variables, 2 performance objectives, i.e., forward voltage gain (S21) and current consumption (IDD), and 17 constraints, defined in Tables 5.12 and 5.13.

Table 5.12 LNA variable ranges

Variable		Min.	Grid resolution	Max.
od1, od2, oddeg (μm)	Outer diameters of the inductors	75	0.01	300
l1, l2, l3, l5 (μm)	Length of the NMOS	0.12	0.1	0.92
gfw1, gfw2, gfw3, gfw5 (μm)	Finger width of the NMOS	1.4	0.1	7.2
nf1, nf2, nf3, nf5	Finger number of the NMOS	1	2	16
lcin, lcdc, lcout (μm)	Length of the capacitors	1	0.1	100
wcin, wcdc, wcout (μm)	Width of the capacitors	1	0.1	100

Table 5.13 LNA Specifications

Specifications	Measure	Target	Units	Description
Objectives	S21	Maximize	dB	Forward voltage gain
	IDD	Minimize	A	Current consumption
Constraints	IDD	≤ 60	mA	Current consumption
	S11@7.9 GHz	≤ -12	dB	Input voltage reflection coefficient
	S12@7.9 GHz	≤ -30	dB	Reversed voltage gain
	S21@7.9 GHz r	≥ 14	dB	Forward voltage gain
	S22@7.9 GHz	≤ -12	dB	Output voltage reflection coefficient
	NF@7.9 GHz	≤ 3	dB	Noise figure
	NFMIN@7.9 GHz	≤ 3	dB	Minimum noise figure
	ZINre@7.9 GHz	≥ 50	Ω	Input impedance (real)
	ZINmag@7.9 GHz $\geq$ 50 Ω	≥ 50	Ω	Input impedance (imag.)
	ZOUTre@7.9 GHz	≥ 50	Ω	Output impedance (real)
	ZOUTmag@7.9 GHz $\geq$ 50	≤ 50	Ω	Output impedance (imag.)
	ZINreMIN	≥ 0	Ω	Min Input impedance
	ZOUTreMIN	≥ 0	Ω	Min output impedance

The optimization variables are the devices' widths and lengths and the outer diameter of the inductors. The biasing devices not identified in the schematic have fixed sizes and are not optimized. The circuit was again, optimized with and without the gradient model for exactly the same conditions, using RF components from the UMC 130 nm design kit.

The optimization was executed for two scenarios, first, for typical conditions, and then, for more aggressive constraints considering the corner conditions described in Table 5.14.

In both cases, whose resulting POFs are shown in Figs. 5.14 and 5.15, the gradient model clearly outperforms the plain simulator in the loop AIDA-C, by

Table 5.14 Considered corner cases conditions

Corner	Description
TYPICAL	VDD is 1.2 V and temperature is 25°
VDD_H_TEMP_H.	VDD is 1.32 V and temperature is 55°
VDD_H_TEMP_L	VDD is 1.32 V and temperature is 0°
VDD_L_TEMP_H	VDD is 1.08 V and temperature is 55°
VDD_L_TEMP_L	VDD is 1.08 V and temperature is 0°
SS	Slow NMOS and slow PMOS
SNFP	Slow NMOS and fast PMOS
FNSP	Fast NMOS and slow PMOS
FF	Fast NMOS and Fast PMOS

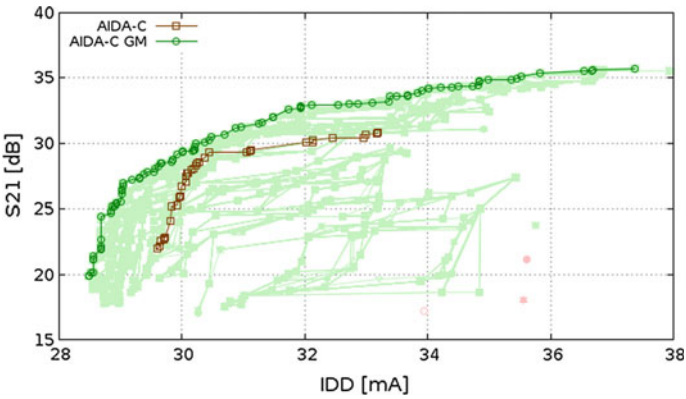


Fig. 5.14 LNA Pareto fronts considering the typical specifications

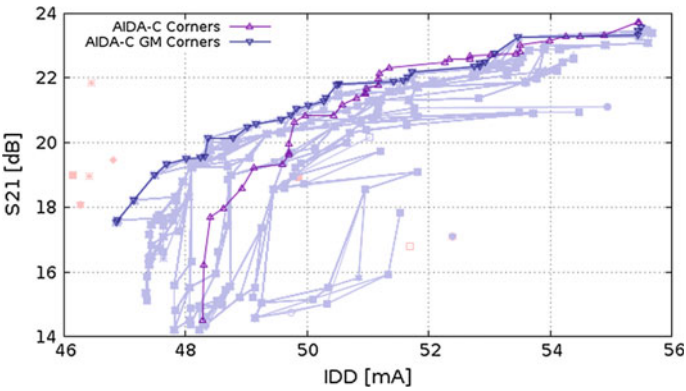


Fig. 5.15 LNA Pareto fronts considering the corner specifications

achieving Pareto fronts with larger area, larger number of solutions and dominant solutions. Notice that in both figures, the fronts illustrated with lighter colors show the Pareto fronts obtained during the optimization process with the gradient model.

Table 5.15 LNA worst case measures for corner results

Measures	Min. IDD	Balanced	Max. S21
IDD (mA)	46,856	50,022	55,494
S11 (dB)	−12,000	−12,190	−12,014
S12 (dB)	−52,930	−51,294	−49,558
S21 (dB)	17,519	21,123	23,579
S22 (dB)	−12,625	−13,475	−16,351
Re(ZIN) (Ω)	50,022	50,176	52,666
Mag(ZIN) (Ω)	55,395	52,284	54,385
min(ZINre) (Ω)	2,576	3,397	4,358
ZOUTre (Ω)	57,478	55,887	50,618
Mag(ZOUT) (Ω)	57,768	55,978	50,881
min(ZOUTre) (Ω)	22,981	22,029	22,690
NoiseFigure (dB)	2,705	2,525	2,435
min(NFMIN) (dB)	1,814	1,670	1,692

From the optimization using the gradient model with corners, three solutions were selected and their measures are presented in Table 5.15. The presented elements contain the maximum S21 solution, the minimum IDD solution and an intermediate solution with balanced S21 and IDD measures.

Despite not being neither an objective or a constraint, the obtained solutions show very good input and output impedance matching. In the maximum S21 case, the real part of both input and output impedances are approximately equal to 50 Ω. Since the value of both the resistive load and the series resistance of the input source is 50 Ω, a perfect input and output impedance matching is almost achieved, which is a fundamental condition to maximize the S21 scattering parameter.

The previous results show that embedding statistical knowledge from the simple but effective automatically generated Gradient Model, into the evolutionary optimization kernel, accelerated the circuit sizing procedure. The model relates the variations of the circuit parameters to the variation of the circuit performance enabling "smart" moves by the optimizer. Thus, the solutions move toward the desired objectives more effective, causing a significant reduction in the number of circuit simulations to obtain the same results.

The Gradient Model integrated with the mutation operator of the genetic algorithm proved to be useful to bias the search direction into the most promising direction. The approach was validated with typical analog circuit structures, showing that, by enhancing the circuit sizing evolutionary kernel with the gradient model, the optimal solutions are achieved, considerably, faster and with identical or superior accuracy.

5.4 Comparison of the Rival Kernels for Analog IC Sizing Optimization

First, the algorithms were experimented and their parameters tuned with constrained problems from CEC 2009 [2] competition. The usage of these benchmarks has two great benefits, first, the optimal front is known, and second, the evaluation is instantaneous, allowing an efficient characterization of the different algorithms tested. In the benchmarks, MOSA showed the best performance.

Then, two analog differential amplifier circuits and a voltage controlled oscillator (VCO) were optimized to study the optimization methods' performance. The study was done considering a fixed number of evaluations for each circuit (circuit simulations) to provide a fair ground for the comparison of the different optimization strategies. Also, for each circuit, two combinations of the number of elements in the population and the number of iterations were considered, (Runset I and Runset II). All runsets are composed by ten independent executions, using a different seed in the random number generator for each run, taken from a common set of seeds. i.e., the same seed was used for the same run of the different algorithms. One additional run with 1,280,000 evaluations was executed (using only NSGA-II) considering a larger population (256) and more generations (5,000) to find a closer estimate of the true POF, giving a reference to assess the previously obtained solutions' quality.

5.4.1 Single-Stage Amplifier with Gain Enhancement Using VCs

The Single-Stage Amplifier with Gain Enhancement using VCs introduced in Sect. 5.1.2 of this Chapter is here used for nominal-only sizing loaded with 6 pF all capacitive load. A common tradeoff in the design of amplifiers is between current (power) consumption and bandwidth (speed). Traditionally these conflicting objectives are reflected in the Figure-of-Merit (FOM) shown in (6.1), where GBW is the gain-bandwidth product, C_{load} is the load capacity and I_{DD} is the current consumption. This FOM is commonly used by designers to access the energy efficiency of the achieved solution. By maximizing the FOM, the power consumption is minimized and the bandwidth product is maximized.

$$FOM = \frac{GBW \times C_{load}}{I_{DD}} \left[\frac{MHz \times pF}{mA}\right] \tag{5.7}$$

Taking advantage of AIDA-C being able to explicitly explore runs the tradeoffs. The design targets are set to maximize GBW and minimize IDD. This option explores the design tradeoffs while implicitly optimizing the FOM. The total amount of simulations in this test-case was 64,000, and the two combinations of the

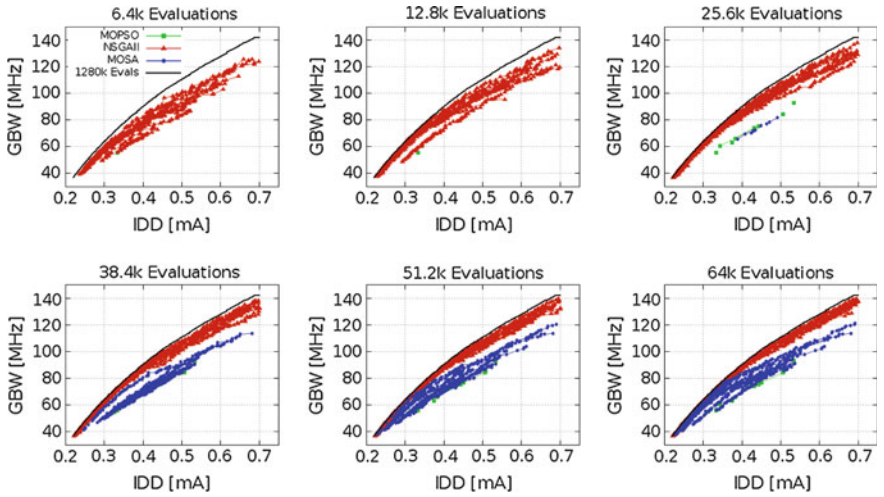


Fig. 5.16 Pareto fronts for the different runs of NSGA-II, MOPSO and MOSA for Runset I

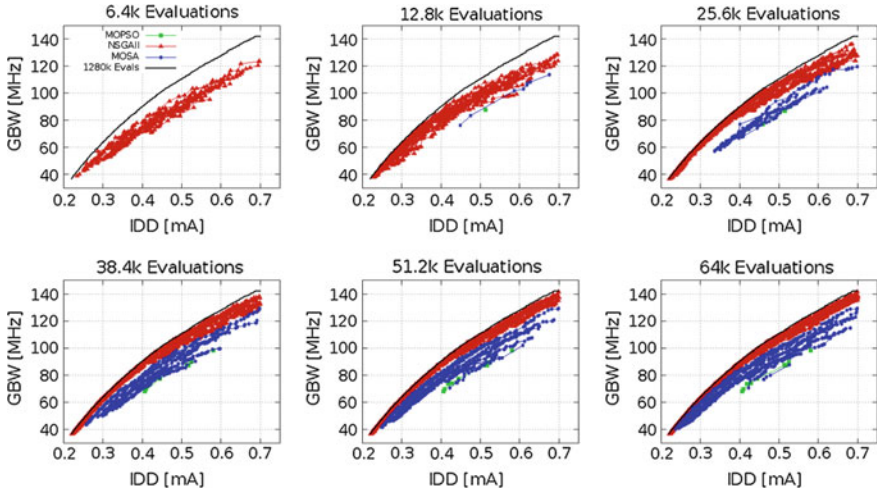


Fig. 5.17 Pareto fronts for the different runs of NSGA-II, MOPSO and MOSA for Runset II

number of elements and the number of iterations considered were, in Runset I, {64, 1000}, and, in Runset II, {128, 500}, respectively. Figure 5.16 shows the Pareto fronts for Runset I at different stages of the optimization process, and Fig. 5.17 shows the same or Runset II.

The evolution of the best FOM, bandwidth and current consumption with the number of simulations for each optimization kernel in both runsets is illustrated in Fig. 5.18.

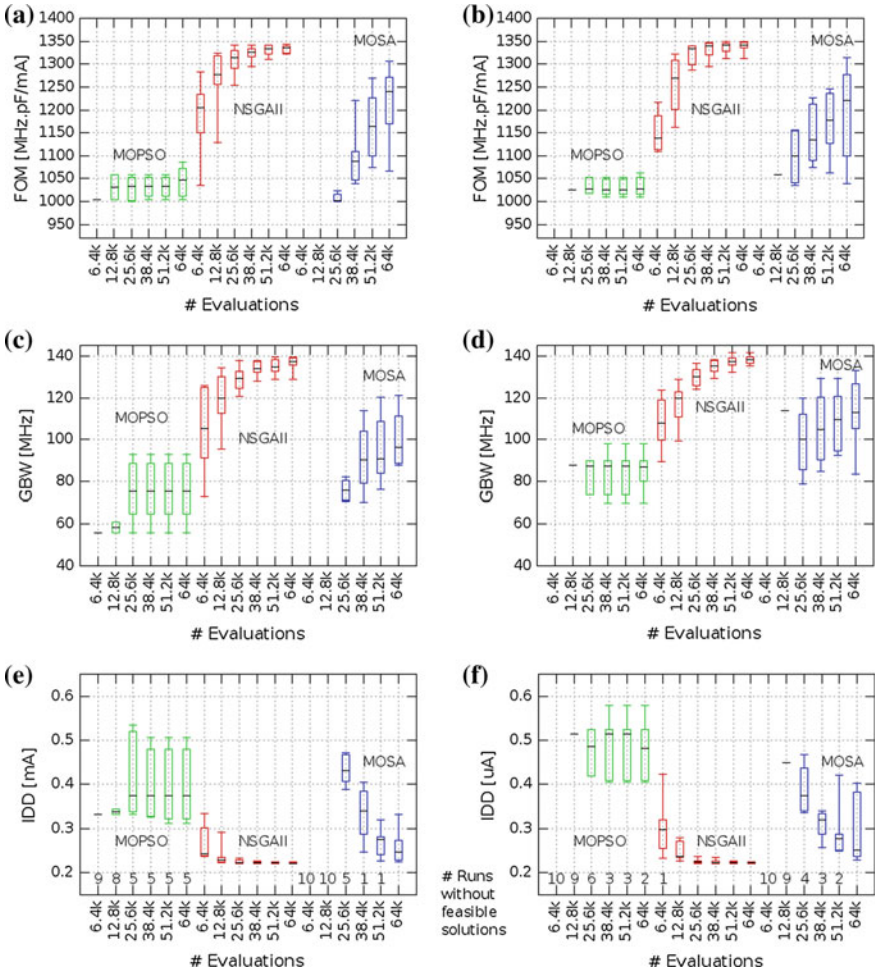


Fig. 5.18 Evolution of the best FOM, GBW, and IDD with the number of simulations for the 10 run. **e** and **f** show the number of runs without feasible solutions. **a** Runset I—best figure of merit. **b** Runset II—best figure of merit. **c** Runset I—best bandwidth. **d** Runset II—best bandwidth. **e** Runset I—best power of consumption. **f** Runset II—best power of consumption

The analysis of the results shows that NSGA-II is much more efficient than MOPSO or MOSA, requiring fewer simulations to achieve the same solutions, and it is more consistent throughout the 10 run. Besides the difference in the obtained solutions is also important to note that NSGA-II consistently get feasible solutions since very early in the optimization process, while the other methods struggle to find feasible solutions. Many of the MOPSO runs did not get any feasible solution in the 64 k evaluations, MOSA performed better, but still requiring a considerably larger number of evaluations when compared to the NSGA-II.

Comparing the performance between Runsets, NSGA-II behaves similarly in both cases, reaching close to the "true" POF, MOPSA struggles in both cases, and MOSA get closer to the "true" POF in Runset II shortening the gap in the high frequency side of the POF.

Please note that, as it can be seen in Fig. 5.18a where from 51.2 to 64 k evaluations of the MOSA the worst FOM actually gets worse, this happens because the runs without feasible solutions, whose count is presented in Fig. 5.18e, f over the x-axis, are not accounted to derive the boxplots, when one or more of the runs that did not had any feasible solutions before, now have a new feasible solution, that new solutions may worsen the worst performance.

5.4.2 *Two-Stage Miller Amplifier*

The other differential amplifier topology considered in this work is the 2-stage Miller operational amplifier introduced in Chap. 3 of this book, whose schematic is reshown in Fig. 5.19 for convenience loaded with a 10 MΩ resistor in parallel with 1 pF capacitor and biased with a current of 10 µA.

The optimization variables are the width, length, and number of fingers of the MOS devices, and the length and number of fingers of the MOM capacitor. The variable ranges are indicated in Table 5.16. Like before, the circuit's performance figures are measured from the simulation results. Table 5.17 indicates the design specifications for this circuit working as a versatile low power DC buffer.

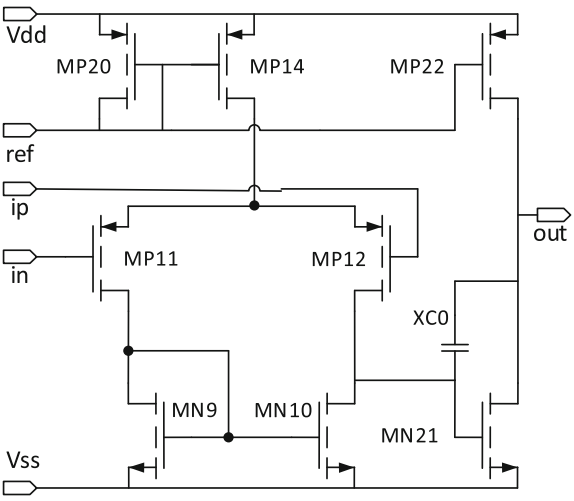


Fig. 5.19 2-stage Miller amplifier schematic

Table 5.16 Variables and ranges for the 2-stage Miller amplifier optimization

Variable (Unit)	Min.	Grid unit	Max.
l1, l4, l6, l8 (nm)	120	5	1,000
w1 w4 w6 w8 (μm)	1	0.1	10
nf1, nf2, nf3, nf4, nf6, nf8	1	2	200
lc (μm)	4.4	0.1	100
Nfc	14	2	198

The variables lc and nfc are the length and number of fingers of the MOM capacitor XC0; l1, w1 and nf1, are the length, width and number of fingers of MP20; l1, w1, and nf2 are the length, width, and number of fingers of M14; l1, w1, and nf3 of MP22; l4, w4, and nf4 of MP11 and MP12; l6, w6, and nf6 of MN9 and MN10; l8, w8, and nf8 of MN21

Table 5.17 Objectives and specifications for the 2-stage Miller amplifier

Specifications	Measure	Target	Description
Objectives	IDD (μA)	Minimize	Current consumption
	GBW (MHz)	Maximize	Unity gain frequency
Constraints	IDD (μA)	≤ 80	Current consumption
	GDC (dB)	≥ 50	Low-frequency gain
	GBW (MHz)	≥ 1	Unity gain frequency
	PM (°)	≥ 55	Phase margin
	PSRR (dB)	≥ 55	Power supply rejection ratio
	SR (V/μs)	≥ 0.8	Slew rate
	Voff (mV)	≤ 1	Offset voltage
	No (μVrms)	≤ 400	Noise RMS
	Sn (nV/$\sqrt{\text{Hz}}$)	≤ 100	Noise density
	OV[1] (mV)	≥ 100	Overdrive voltages $(V_{GS} - V_{TH})^2$
	D[1] (mV)	≥ 50	Saturation margin $(V_{DS} - VD_{Sat})^2$

[1]The constraint applies to: MP11, MP12, MP14, MP20, MN21 and MP22
[2]For PMOS devices the overdrive is Vt − Vgs and delta is Vdsat − Vds

The number of simulations was double in this test-case to further understand the behavior of the methods with more evaluations, leading to a Runset I with 64 elements and 2,000 iterations, and a Runset II with 128 elements and 1,000 iterations. Figure 5.20 shows the Pareto fronts for Runset I at different stages of the optimization process, namely at 6,400, 12,800, 25,600, 51,200, 89,600 and 128,000 simulations, and Fig. 5.21 shows the same for Runset II.

The evolution of the best FOM, bandwidth and current consumption with the number of simulations for each optimization kernel in both runsets is illustrated in Fig. 5.22.

Again, the results show that NSGA-II is more efficient than MOPSO or MOSA, consistently finding better solutions throughout the entire optimization process. Unlike the previous test-case, where both NSGA-II and MOSA manage to almost

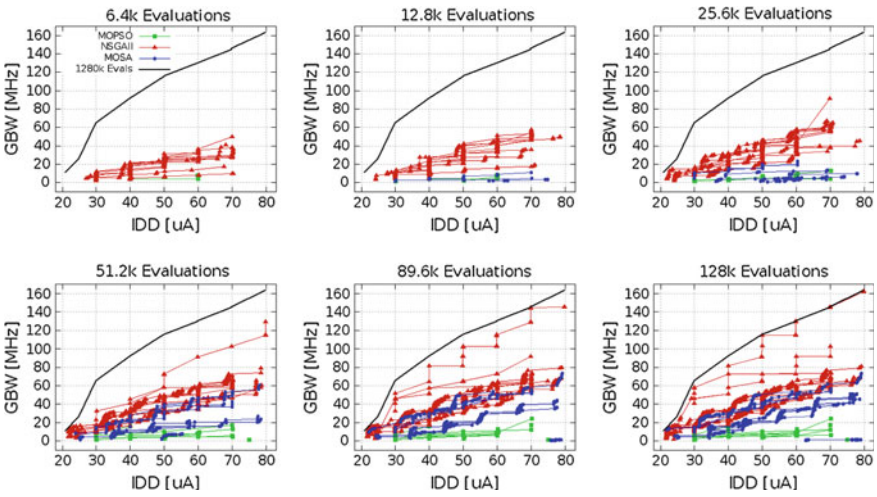


Fig. 5.20 Pareto fronts for the different runs of the NSGA-II, MOPSO and MOSA on the Two-Stage amplifier problem for Runset I

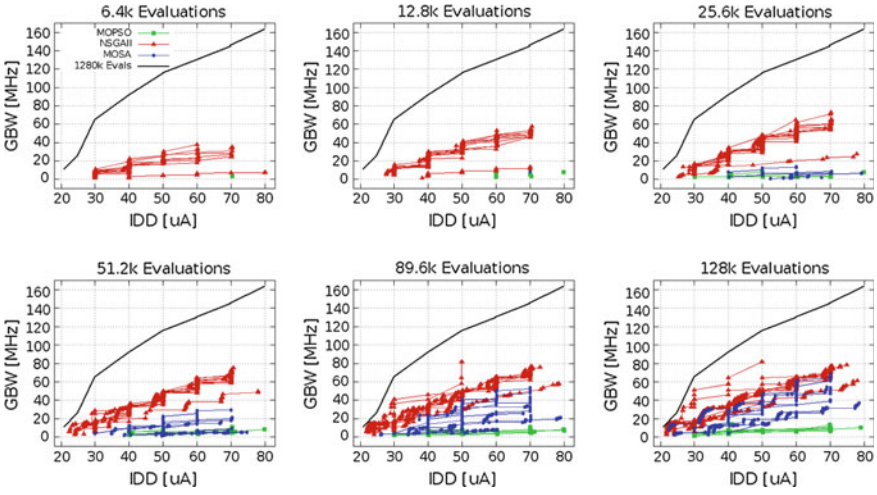


Fig. 5.21 Pareto fronts for the different runs of the NSGA-II, MOPSO and MOSA on Two-Stage amplifier problem for Runset II

reach the "true" POF with 64,000 simulations, and despite the extra simulations considered (twice as much), in this test case all algorithms stayed reasonably far from it. These results show that different circuit or specifications result in very different algorithm performance for similar algorithm configurations in similar search space sizes.

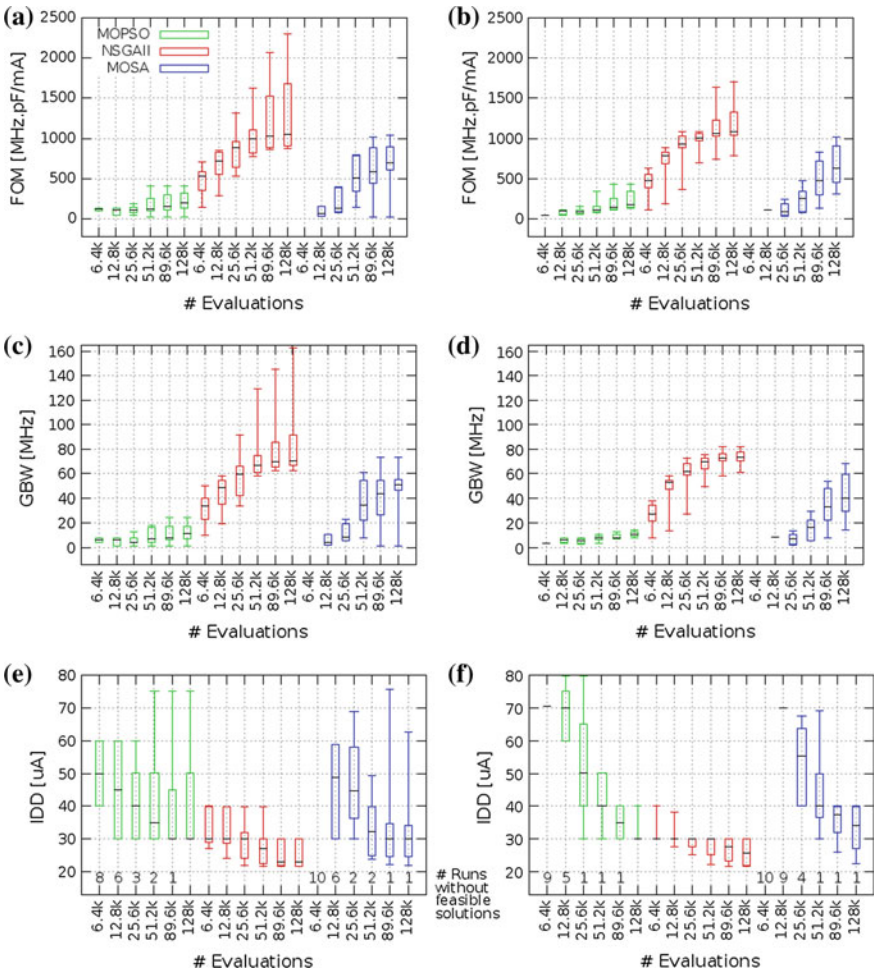


Fig. 5.22 Evolution of the best FOM, GBW, and IDD with the number of simulations in the Two-Stage amplifier runsets. **e** and **f** show the number of runs without feasible solutions. **a** Runset I—best figure of merit. **b** Runset II—best figure of merit. **c** Runset I—best bandwidth. **d** Runset II—best bandwidth. **e** Runset I—best current consumption. **f** Runset II—best current consumption

Also, it is worth mentioning that in this test-case both MOSA and MOPSO struggle to find feasible solutions. Also noticeable was the fact that, even though MOSA persistently took longer to find feasible solutions (and in some runs it did not found any), when feasible solutions were found, it almost manage to catch up with NSGA-II.

In terms of comparing outputs in Runset I with the outputs in Runset II, it is seen that the average results are almost the same. However, two notes; first, in Runset I some runs get closer to the true POF, confirming that more iterations with fewer

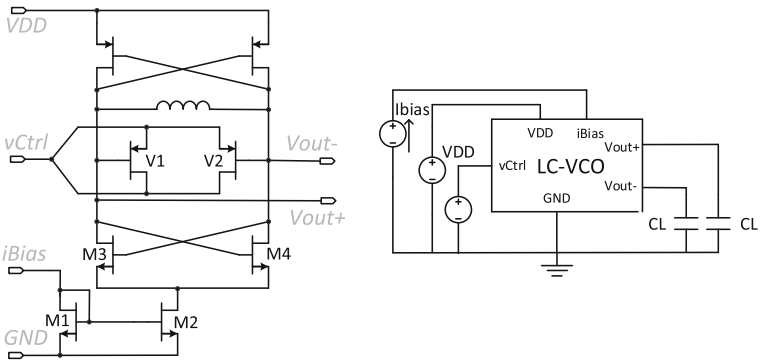


Fig. 5.23 Schematic of the LC-VCO

elements is better to push the quality of the solutions, second also in Runset I it was more difficult to achieve feasibility (this is also true for the Single-Stage amplifier test-case) as fewer elements limit the exploration capabilities of the optimization methods.

5.4.3 LC-Voltage Controlled Amplifier

The LC-VCO circuit from [3], shown in Fig. 5.23, with a 1 pF load capacitance was optimized with the implemented multi-objective optimization kernels. In general when designing oscillators, the target it to minimize both phase noise and power consumption for a given oscillation frequency. These conflicting objectives are reflected in the Figure-of-Merit (FOM) defined by Kinget [4] and shown in (5.8), where ω_0 is the oscillation frequency, P_{dc} is the power consumption, $\Delta\omega$ is the offset from the output frequency and $L(\Delta\omega)$ is the oscillator phase noise, given by (5.9), the original Leeson's equation [5], where Q is the loaded quality factor of the oscillator, k is the Boltzmann's constant, T is the absolute temperature, P_{sig} is the oscillation output power, F is the noise factor of the amplifier and $\Delta\omega_{1/f3}$ is the corner frequency between $\omega_{1/f2}$ and $\omega_{1/f3}$ portion of the phase noise spectrum [6].

$$\text{FOM} = L(\Delta\omega) + 10\log\left[\frac{P_{\text{dc}}}{1\text{mW}} \times \left(\frac{\Delta\omega}{\omega_0}\right)^2\right] \tag{5.8}$$

$$L(\Delta\omega) = \frac{P(\Delta\omega)}{P(\omega_0)} = 10\log\left\{\frac{2FkT}{P_{\text{sig}}} \times \left[1 + \left(\frac{\omega_0}{2Q\Delta\omega}\right)^2\right] \times \left(1 + \frac{\Delta\omega_{1/f3}}{|\Delta\omega|}\right)\right\} \tag{5.9}$$

Table 5.18 Objectives and specifications for the LC-VCO design

Specifications	Measure	Target	Units	Description
Objectives	PN	Minimize	dBc/Hz	Phase noise measured @ 1 MHz
	P	Minimize	mW	Power consumption
Constraints	IDD	≤ 6	mW	Power consumption
	OF	≥ 2.4; ≤ 2.4835	GHz	Oscillation frequency
	OVS	≥ 100	mV	Output signal amplitude
	PN	≤ -100.0	dBc/Hz	Phase noise @ 1 MHz
	OV[1]	≥ 80	mV	Overdrive voltages $(V_{GS} - V_{TH})^2$
	D[1]	≥ 50	mV	Saturation margin $(V_{DS} - V_{DSat})^2$

[1]The constraint applies to: M1, M2, M3, M4, M5 and M6
[2]For PMOS devices the overdrive is Vt − Vgs and delta is Vdsat − Vds

Table 5.19 Variables and ranges for the LC-VCO design

Variable (Unit)	Min.	Grid resolution	Max.
l_1, l_3, l_5, l_{var} (nm)	120	10	920
w_1, w_3, w_5, w_{var} (μm)	1.0	0.1	10.0
nf_1, nf_3, nf_{var}	4	2	16
nf_5	4	2	32
outd (μm)	90.00	0.01	290.00
ib (mA)	0.1	0.1	5.0

The variables l_1, w_1, and nf_1 are dimensions of M1 and M2; l_3, w_3, and nf_3 of M3 and M4; l_5, w_5 and nf_5 of M5 and M6; l_{VAR}, w_{VAR} and nf_{VAR} of V1 and V2; *outd* is the outer diameter of the inductors, and *ib* the value of the bias current

This FOM is commonly used by designers to assess the quality of the achieved solution, where the smaller its value, the better the performance of the corresponding oscillator.

In this test case, the LC-VCO is designed to for the previously introduced design objectives of minimum both phase noise (measures at 1 MHz) and power consumption. The complete specifications are shown in Table 5.18. Table 5.19 shows the design variables and their ranges.

The total amount of simulations in each test run was 64,000, and the two combinations of the number of elements and the number of iterations considered, were respectively, {64, 1000} in Runset I, and {128, 500} in Runset II. Figure 5.24 shows the Pareto fronts for Runset I at different stages of the optimization process, namely at 6,400, 12,800, 25,600, 38,400, 51,200 and 64,000 simulations, while Fig. 5.25 shows the same for Runset II. The progression of the best power, phase noise and FOM found with the number of simulations for each optimization kernel in both runsets is illustrated in Fig. 5.26.

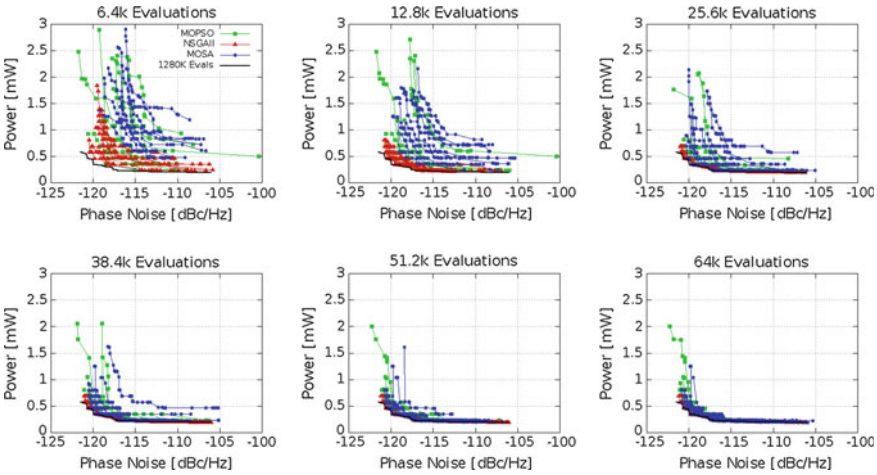


Fig. 5.24 Progression of the Pareto front with the number of simulations for the different runs of the NSGA-II, MOPSO and MOSA for Runset I

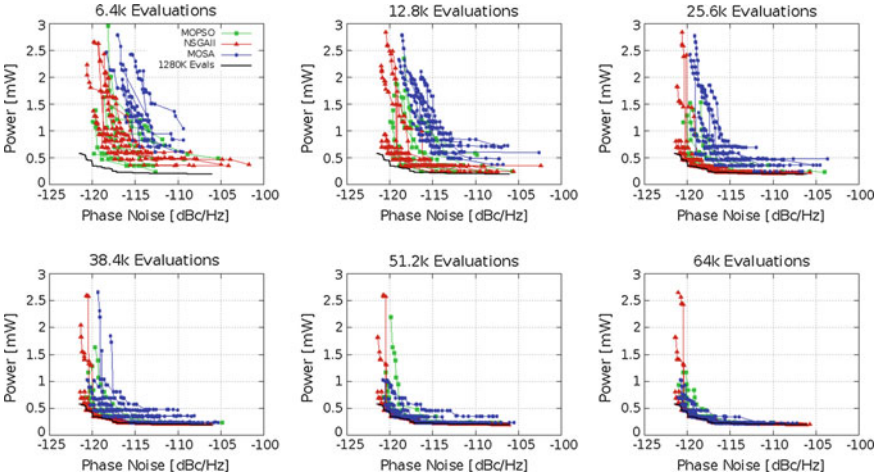


Fig. 5.25 Progression of the Pareto front with the number of simulations for the different runs of the NSGA-II, MOPSO and MOSA for Runset II

The results show, yet again, that NSGA-II is much more efficient than MOPSO or MOSA, requiring fewer simulations to reach the same solutions and being consistent throughout the 10 run. Nevertheless and, unlike the amplifier circuits, in the LC-VCO design, even using the other algorithms, after a couple of hours, multiple LC-VCO are designed showing state-of-the-art FOMs.

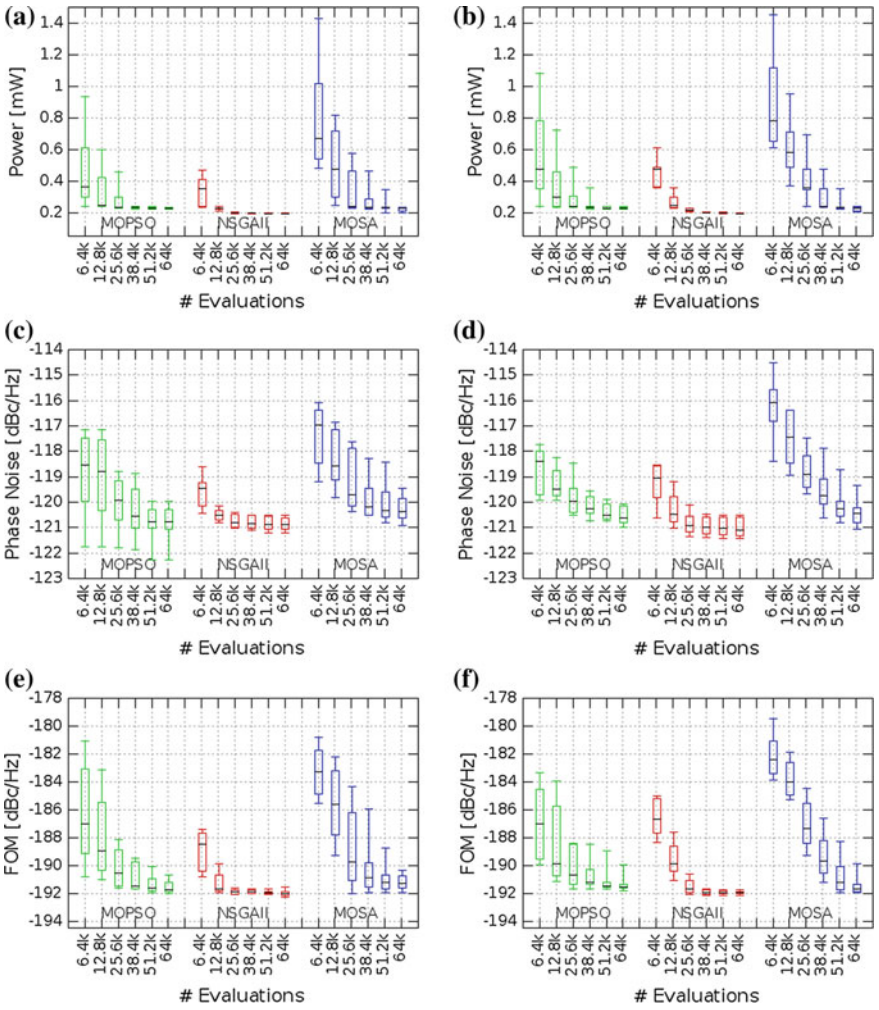


Fig. 5.26 Evolution of the best P, PN, and FOM with the number of simulations in the LC-VCO runsets. **a** Runset I—best power consumption. **b** Runset II—best power consumption. **c** Runset I—best phase noise. **d** Runset II—best phase noise. **e** Runset I—best figure of merit. **f** Runset II—best figure of merit

From the designer perspective, this multi-objective design approach outputs multiple design with very good FOM, allowing for an effective exploration of the design tradeoffs. But process and environment variations should be accounted in the design for realistic comparison with other works.

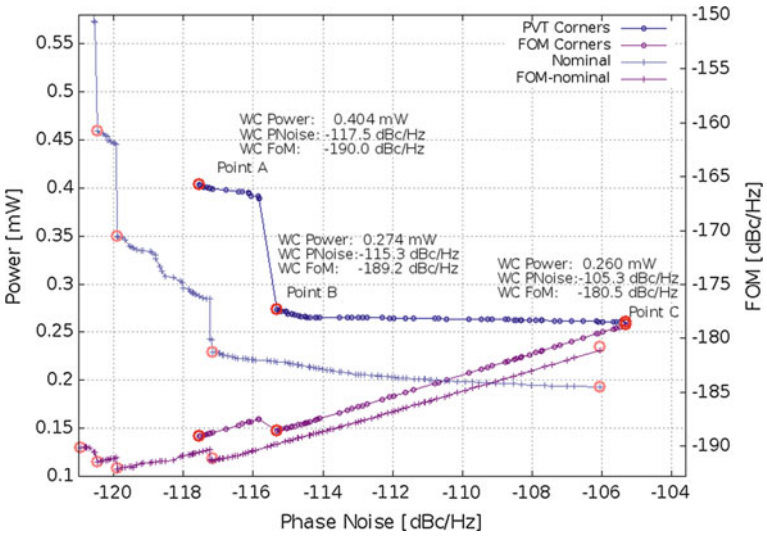


Fig. 5.27 LC-VCO multi-objective optimization result for corner conditions

To account for process and environment variations, the solutions obtained for nominal conditions were re-optimized considering also PVT corners. Namely C1, C2, C3, C4, C5, C6, C7 and C8 corresponding to 1.32 V/55 °C, 1.32 V/0 °C, 1.08 V/55 °C, 1.08 V/0 °C, SS, SF, FS and FF corners behavior, respectively. In Fig. 5.27, the POF obtained considering corner conditions is presented and shows some performance degradation on the solutions due to worst case (WC) considerations.

The solution points, considering the corner conditions, marked in the plot are detailed in Table 5.20. The first line on each solution corresponds to the behavior of the sizing solution, when subject to nominal working conditions, the worst values from the corner simulations are in bold.

From Table 5.20, is possible to observe that, despite the performance degradation due to process, voltage and temperature variations, the values corresponding to an extremely robust solution are still quite compelling, namely in terms of FOM, PN and power consumption. Table 5.21 summarizes the "best" results obtained in this case study, while comparing them to other published works. There it can be seen that they achieve a competitive FOM when compared with other state-of-the-art LC-VCOs, while explicitly and efficiently exploring the design tradeoff between phase noise and power consumption in a single run of the tool.

Table 5.20 Solutions details for all corner cases

		Freq. (GHz)	Power (mW)	Δf (MHz)	L(δf) (dBc/Hz)	|FoM| (dBc/Hz)
Point A Lowest phase noise		2.43	0.357	1.0	−119.3	−191.5
	C1	2.44	**0.404**		−118.3	**−190.0**
	C2	2.44	0.399		−119.6	−191.3
	C3	2.42	0.315		**−117.5**	−190.2
	C4	2.43	0.309		−119.4	−192.2
	C5	2.43	0.357		−119.3	−191.5
	C6	2.45	0.354		−119.1	−191.4
	C7	2.40	0.360		−119.5	−191.5
	C8	2.47	0.379		−118.1	−190.2
Point B		2.45	0.243	1.0	−117.0	−190.9
	C1	2.45	**0.274**		−115.8	**−189.2**
	C2	2.46	0.271		−117.9	−191.4
	C3	2.44	0.215		**−115.3**	−189.7
	C4	2.45	0.212		−117.9	−192.4
	C5	2.45	0.243		−117.0	−190.9
	C6	2.47	0.241		−116.9	−191.0
	C7	2.43	0.245		−116.9	−190.7
	C8	2.48	0.254		−115.5	−189.4
Point C Lowest power		2.45	0.214	1.0	−113.5	−187.9
	C1	2.46	**0.260**		−112.5	−186.1
	C2	2.46	0.249		−115.9	−189.8
	C3	2.43	0.180		**−105.3**	**−180.5**
	C4	2.44	0.168		−112.7	−188.3
	C5	2.45	0.214		−134.6	−187.9
	C6	2.47	0.209		−133.3	−188.0
	C7	2.43	0.219		−135.8	−187.9
	C8	2.48	0.252		−114.1	−188.0

5.5 Conclusion

The proposed approach was validated with both classical and new analog circuit structures showing its generality. First the impact of the evolutionary parameters was study, followed by an analysis of the impact of also considering the constraints derived from the worst case corners. By enhancing the evolutionary kernel with the gradient model, the optimal solutions are achieved, considerably, faster and with identical or superior performance. The performance of three rival optimization algorithms (NSGA-II, MOPSO and MOSA) was compared. The automatic sizing and optimization obtained various feasible solutions with each algorithm, and a

Table 5.21 Summary of the most competitive LC-VCO solutions, showing other SOA results

References		Tech. (nm)	Freq. (GHz)	Power (mW)	Δf (MHz)	L(Δf) (dBc/Hz)	FoM (dBc/Hz)
[7][1]		90	2.42	0.515	1.0	−119.7	−190.3
[8][2]		180	2.4	0.60	1.0	−103.7	−173.5
[9][2]		180	2.02	3.15	0.6	−118.9	−186.0
[10][2]		180	2.63	0.432	0.4	−106.4	−186.4
[11][2]		90	2.16	0.528	0.4	−106.2	−183.6
[12][2]		90	2.40	0.564	1.0	−116.8	−186.9
[12][2]		130	2.40	0.624	1.0	−117.1	−186.8
[13][2]		180	2.00	1.0	0.1	−103.0	−189.0
[14][2]		180	2.74	5.4	3	−135.0	−186.9
[15][2]		180	2.62	3.2	1	−127.1	−190.4
[16][2]		180	2.30	1.5	1	−120.0	−185.4
This work[3]	PA	130	2.40	0.404	1.0	−117.5	−190.0
	PB			0.274		−115.3	−189.2
	PC			0.260		−105.3	−180.5

[1]Simulation
[2]Measurement
[3]Simulation with corners

comparison between the obtained solutions with other recently published works was made, proving the potential of the optimization tool, as well as the potential of the optimization algorithms.

References

1. Póvoa R, Lourenço N, Horta N, Santos-Tavares R, Goes J (2013) Single-stage amplifiers with gain enhancement and improved energy-efficiency employing voltage-combiners. In: 21st IFIP/IEEE international conference on very large scale integration, Istambul, 2013
2. Zhang Q, Zhou A, Zhao S, Suganthan PN, Liu W, Tiwar S (2009) Multiobjective optimization test instances for the CEC 2009 special session and competition. http://dces.essex.ac.uk/staff/zhang/MOEAcompetition/cec09testproblem0904.pdf. Accessed 17 March 2016
3. Póvoa R, Lourenço R, Lourenço N, Canelas A, Martins R, Horta N (2014) LC-VCO automatic synthesis using multi-objective evolutionary techniques. In: 2014 IEEE international symposium on circuits and systems (ISCAS), Melbourne VIC, 2014
4. Kinget P (1999) "Integrated GHz voltage controlled oscillators". In: Analog circuit design, Springer, US, pp 353–381
5. Leeson DB (1966) A simple model of feedback oscillator noise spectrum. IEEE 54 (2):329–330
6. Lee T (1998) The design of CMOS radio-frequency integrated circuits. Cambridge University Press, Cambridge

7. Rout PK, Nanda UK, Acharya DP, Panda G (2012) Design of LC VCO for optimal figure of merit performance using CMODE. In: 1st International conference recent advances in information technology, Dhanbad, 2012
8. Perumana B, Mukhopadhyay R (2008) A low-power fully monolithic subthreshold CMOS receiver with integrated LO generation for 2.4 GHz wireless PAN applications. IEEE J Solid-State Circuits 43(10):2229–2238
9. Lin J, Jian-Guo M, Kiat-Seng Y, Do A (2005) A novel methodology for the design of LC tank VCO with low phase noise. In: IEEE international symposium on circuits and systems, Kobe, 2005
10. Lee H, Mohammadi S (2007) A subthreshold low phase noise CMOS LC VCO for ultra low power applications. IEEE Microwave Wirel Compon Lett 17(11):796–798
11. Fiorelli R, Peralias E, Silveira F (2011) LC-VCO design optimization methodology based on the gm/ID ratio for nanometer CMOS technologies. IEEE Trans Microw Theory Tech 59 (7):1822–1831
12. Siwiec K, Borejko T, Pleskacz W (2012) LC-VCO design automation tool for nanometer CMOS technology. In: IEEE 15th international symposium on design and diagnostics of electronic circuits and systems, Tallinn, 2012
13. Yun S-J, Shin S-B, Choi H-C Lee S-G (2005) A 1 mW current-reuse CMOS differential LC-VCO with low phase noise. In: Digest of technical papers IEEE international solid-state circuits conference 2005 (ISSCC. 2005), San Francisco, CA, 2005
14. Jia L, Ma JG, Yeo KS, Yu XP, Do MA, Lim WM (2006) A 1.8-V 2.4/5.15-GHz dual-band LCVCO in 0.18-μm CMOS technology. IEEE microwave and wireless components letters 16 (4):194–196
15. Jain S, Jang SL, Tchamov NT (2016) Tuned LC-Resonator Dual-Band VCO. IEEE Microwave Wirel Compon Lett 26(3):204–206
16. Kao HL, Yang DY, Kao CH, Chang YC, Lin BS (2008) Switched resonators using adjustable inductors in 2.4/5 GHz dual-band LC VCO. Electron Lett 44(4):299–300

Chapter 6
Layout-Aware Circuit Sizing

6.1 Motivation for Layout-Aware Circuit Sizing

Traditionally, the layout generation task, where the sized electronic devices are drawn from the sizes obtained during the sizing task in the set of 2D masks used in the fabrication of the integrated circuits (ICs), is only triggered when the sizing task is complete. However, layout induced parasitic effects strongly affect the performance of analog circuits, and to achieve post-layout successful designs that meet all specifications, time-consuming and non-systematic iterations between these electrical and physical design phases, as illustrated on Fig. 6.1, are required. Without them, performance overdesign during sizing results in wasted power and area, and performance underdesign may compromise the circuits' post-layout performances [1, 2]. Furthermore, accurate evaluation of some layout related measures, like circuit area or aspect ratio is practically impossible from the netlist alone. Thus, to address post-layout performance degradation earlier in the design flow, sizing and layout design phases tend to overlap [2].

By having the sizing task sensitive to layout induced effects, iterations between sizing and layout are reduced. Layout-aware/layout-inclusive sizing methodologies aim at obtaining a design flow that avoids the time consuming iterations between circuit and physical synthesis, by bringing layout-related data into the sizing process.

Despite the advantages of layout-aware methodologies, both layout generation and extraction in each iteration of the sizing loop are expensive and hard to setup operations. The layout generation for accurate parasitic extraction is the bottleneck of layout-aware methodologies. Some approaches rely on parametric cell generators [2–4] that code the whole layout of a circuit, which lack on flexibility and requires long/complex setup before each design. The alternative is fully automatic generation, as proposed in [5], that avoids parametric generators, but still does not automatically update the routing topology to properly fit the multitude of devices' sizes/shapes and performances that are explored on the search for the final solution,

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0_6

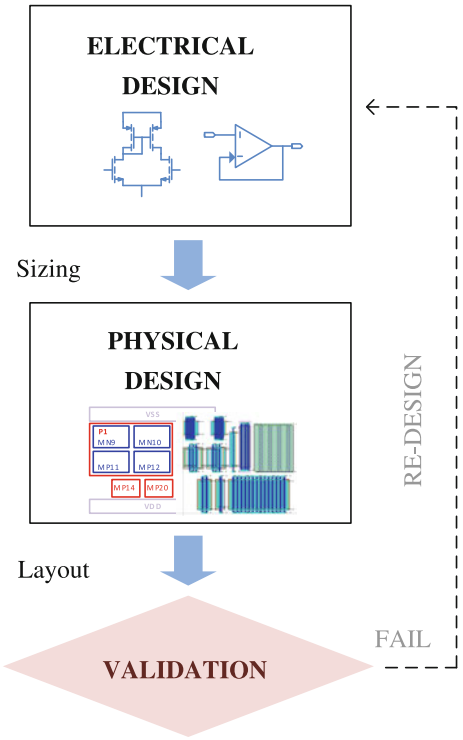


Fig. 6.1 Traditional design flow: iterations between electrical and physical design phases

and also, takes hours for a few iterations of the optimization engine. This time cost still makes the use of automatic layout generators inside the sizing optimization loop impractical.

Figure 6.2 illustrates different evaluation techniques that are proposed in AIDA's [6, 7]. Optimization-based circuit sizing to increasingly include layout effects. Unlike previous layout-aware works, where the complete layout is generated and evaluated using post-extracted performance values during the optimization [2, 5], in the floorplan-aware flow a set of floorplan constraints providing a general, simple-to-derive, fast-to-compute and accurate floorplan to be used during the circuit optimization. In this approach, instead of time-consuming complete layout generation and extraction to control the parasitic effects, undesirable layout induced effects are alleviated using designers' experience to define, using simple constructs, where and how to lay out the devices. In this way, by ensuring matching, symmetry and proximity constraints on relevant devices [8], the impact of deviations due to the fabrications process, as noted in [9], is reduced.

Aware of the impact of parasitics in the degradation of the performance, this chapter also presents a methodology for automatic layout-aware synthesis of analog ICs that uses a built-in extractor to accurately compute the impact of parasitic from both floorplan and early-stages of routing. The computation cost is reduced by

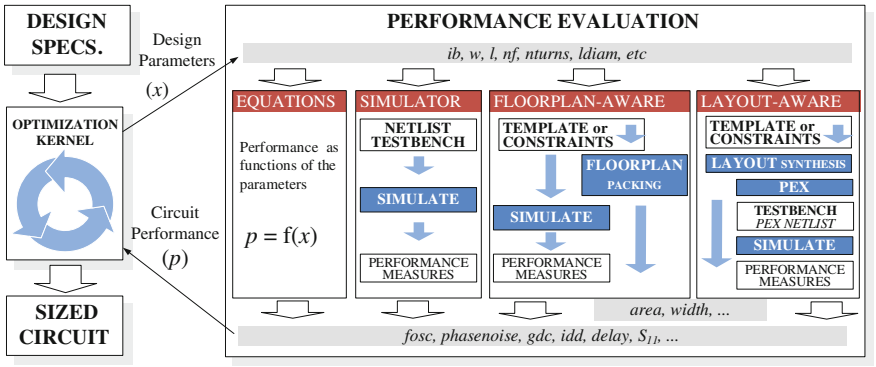


Fig. 6.2 Optimization based layout-aware sizing flow

avoiding the time-consuming detailed layout, needed by off-the-shelf tools for parasitic extraction. Moreover, prior knowledge of the circuit's parasitics is not required, i.e., no circuit specific equations or models are required [3]. The post-layout performances of the circuits designed with the proposed layout-aware methodology, versus, the traditional optimization-based sizing, show the applicability of the approach for simultaneous circuit sizing and layout generation.

In the proposed layout-aware flow with AIDA, the embedded layout generator computes the optimal electrical-current correct wiring topology and global routing in-loop for each different sizing solution, which was never considered on previous approaches to layout-aware electrical synthesis. In this way, not only the geometric requirements of the floorplan and parasitic-aware performances, but also, the routing quality will follow the optimization process. By using a lightweight built-in extractor it is possible to accurately compute the impact of layout parasitics for both floorplan and early-stages of routing, without requiring the detailed final layout (i.e., all design-rules and layout-versus-schematic errors solved). Traditionally, the last step required in the automatic layout generation flow is the detailed routing, which is by far the most computational intensive and time consuming step. By avoiding the need of a detailed layout and also avoiding an external extractor, the overall optimization time is greatly reduced. Is important to note that the internal extractor does not need to be more efficient when compared to the highly optimized commercial extraction tools, but the simplified models and correct-by-construction layout do simplify the implementation. Also, by avoiding inter-process iterations between the optimizer and the external tool the overall runtime is reduced.

Furthermore, this layout-aware flow takes into consideration worst case corners in the circuit-level optimization in order to overcome environment and die-to-die variations earlier in the design flow, and also, entwine both layout-aware and variability-aware time consuming iterations. The designer defined worst case PVT corners are accounted in the performance evaluation. Typically, variability-aware sizing, if considered, is performed prior to the traditional layout generation or to a layout-aware flow. But the speedup achieved in the layout generation and parasitic

extraction without requiring an external extraction tool, allows the inclusion of worst case PVT analysis for robust design over process and environment variations, without burdening the computational time of the layout-aware synthesis. From the best of our knowledge, the two ideas have never been proposed in the same methodology.

6.2 Floorplan-Aware Circuit Sizing

Following a different path from previous layout-aware works, where the complete layout is generated and the circuit is evaluated using post-layout extracted performance values during the optimization [2, 5], in the floorplan-aware circuit sizing a set of floorplan constraints providing a general, reusable, technology independent, simple-to-derive, fast-to-compute and accurate geometric layout estimate is used during the optimization.

In this approach, the floorplan provides accurate evaluation of the geometric requirements of the circuits (area, aspect ratio, etc.), and instead of the time-consuming complete layout generation and extraction to control the parasitic effects, the undesirable layout induced effects are alleviated by the designer using simple constructs to define where and how to lay the devices, enforcing matching, symmetry and proximity constraints on the individual devices. Further matching is also accomplished using complex layout structures like common-centroid or interdigitated available from AIDA's analog module generator (AMG) briefly described in Sect. 6.2.2 of this chapter.

6.2.1 Floorplan-Aware Flow

The floorplan-aware circuit sizing implemented in AIDA-C is shown Fig. 6.3. AIDA-L's floorplanner, which enforce the topological constraints described in the

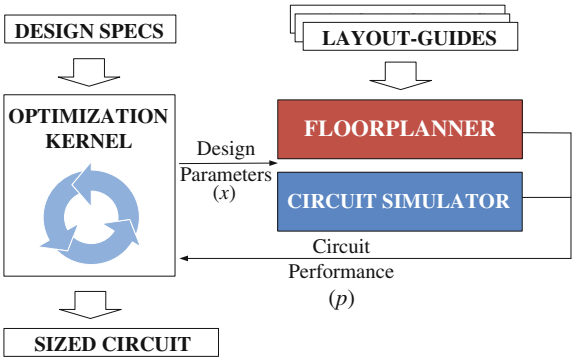


Fig. 6.3 Floorplan-aware evaluation flow

designer-supplied layout guidelines, is included. The floorplanner accurately measures the geometrical properties of the circuit and is used together with the circuit simulator to evaluate the circuit's geometric requirements and performance, respectively, during the optimization loop.

The major steps of the floorplan-aware evaluation are highlighted by order:

1. AIDA-C optimizer sets x, the design variables, (e.g., devices' widths, lengths, number of fingers, etc.) for the tentative sizing solution being evaluated;
2. AIDA-L floorplanner generates multiple floorplans for each tentative solution, and the one that better suits the geometrical requirements is considered to accurately evaluate the geometrical properties of the circuit;
3. The circuit performance is measured from the set of testbenches (DC, AC, TRAN, etc.) using the electrical simulator for the defined PVT corners;
4. Both the geometric and electrical performance measures are used together in the optimization process.

The main disadvantages of traditional templates for layout are the large development time, their maintenance and that they become inadequate for wide parameter changes. In AIDA-C's flow, development and maintenance costs are reduced to a minimum by keeping the layout guides simple and intuitive, using the floorplan description introduced in Chap. 3 of this book, which is easily visualized and edited.

The last issue, the lack flexibility, is especially relevant in a multi-objective system where a multitude of solutions is found even for the same specifications. The Pareto set encompasses very different circuit solutions and a single floorplan would hardly be the best choice for all solutions, enforcing unreasonable topological constraints in many of them. To prevent such over constraining of the search space, not one, but multiple floorplans are considered. The multiple floorplans add increased flexibility to accommodate specification change and allow better packing of the layout throughout the entire solution set.

Figure 6.4 illustrates alternative floorplans that can be obtained for the same sizing of the devices (width and length of the transistors) and shows a few different

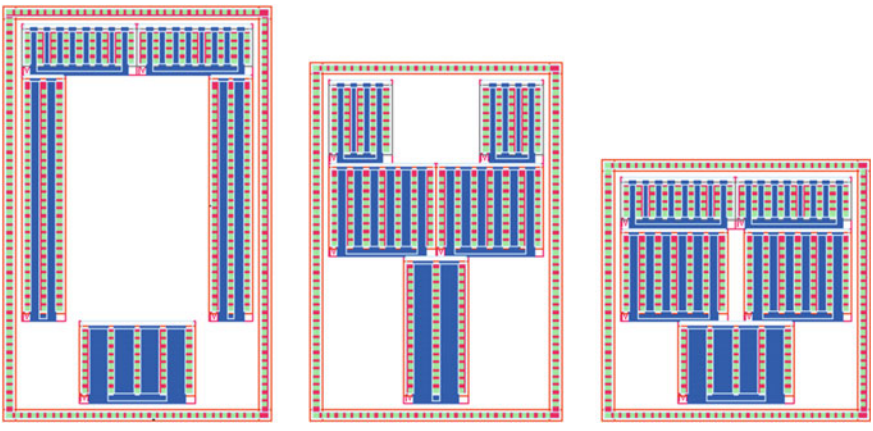


Fig. 6.4 A set of placement solutions for the same template and same sizing

placements for the same template, generated by varying only the number of fingers of the transistors. From this simple example is easy to notice that the "geometrical" device parameters, e.g., number of fingers and number of transistor rows, have a stronger impact in the area than the "design" parameters (width and length).

However, changing geometrical parameters to better fit the layout, modifies device's properties, e.g., the source/drain area and perimeter, the distance from the gates to shallow trench isolation (STI), etc., which have impact in the devices' performance. Despite the lack of parasitic extraction, considering all these parameters when evaluating the circuit effectively shortens the gap between pre- and post-layout circuit simulation [10].

6.2.2 *Analog Module Layout Generator*

Before moving to the description of the floorplanner, a brief overview of the Analog Module Generator [11] is provided. The reason for the inclusion of an AMG in AIDA is the need to efficiently create complex layout structures for the devices.

The AMG is a parametric module generator capable of creating simple and complex structures for a device or group of devices, like folded transistors, common-centroid layout styles of transistors or capacitors, merged structures, etc. [12], and is integrated with both, AIDA-C and AIDA-L. The AMG implemented in AIDA has the structure presented in Fig. 6.5. For each set of technology rules, the AMG creates correct-by-construction layout instances. All the modules are defined in a technology-independent manner, i.e., all the modules are described in terms of

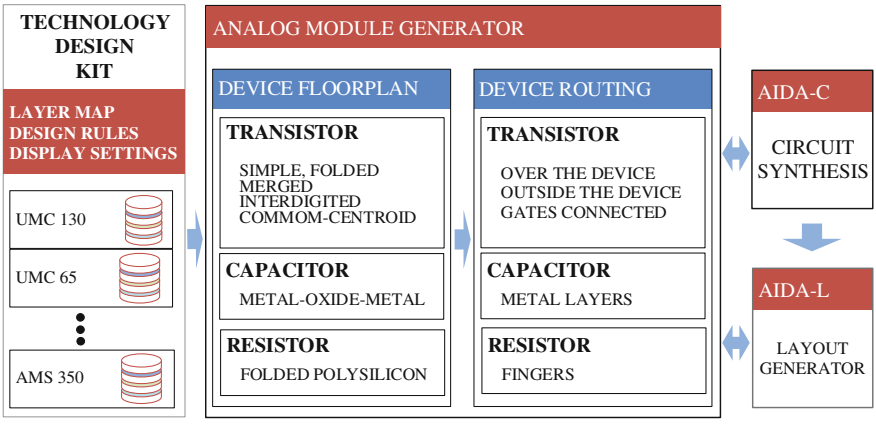


Fig. 6.5 Analog Module Generator architecture

technology design rules which are specified in technology-dependent different set of files, which eases the reuse of constructors between different technologies. The technology definition files are the layer map file, where the GDS layer number is mapped to the AIDA layer identifier; the display settings file, where the information for displaying each layer in the AIDA's graphical user interface is defined; and the design rule file, where the geometric constraints, e.g., minimum distances, encloses or extensions between layers, of the technology are defined.

Figure 6.6 illustrates some transistors produced by the AMG. The basic active device is the folded transistor shown in Fig. 6.6a. Besides folded transistors, where

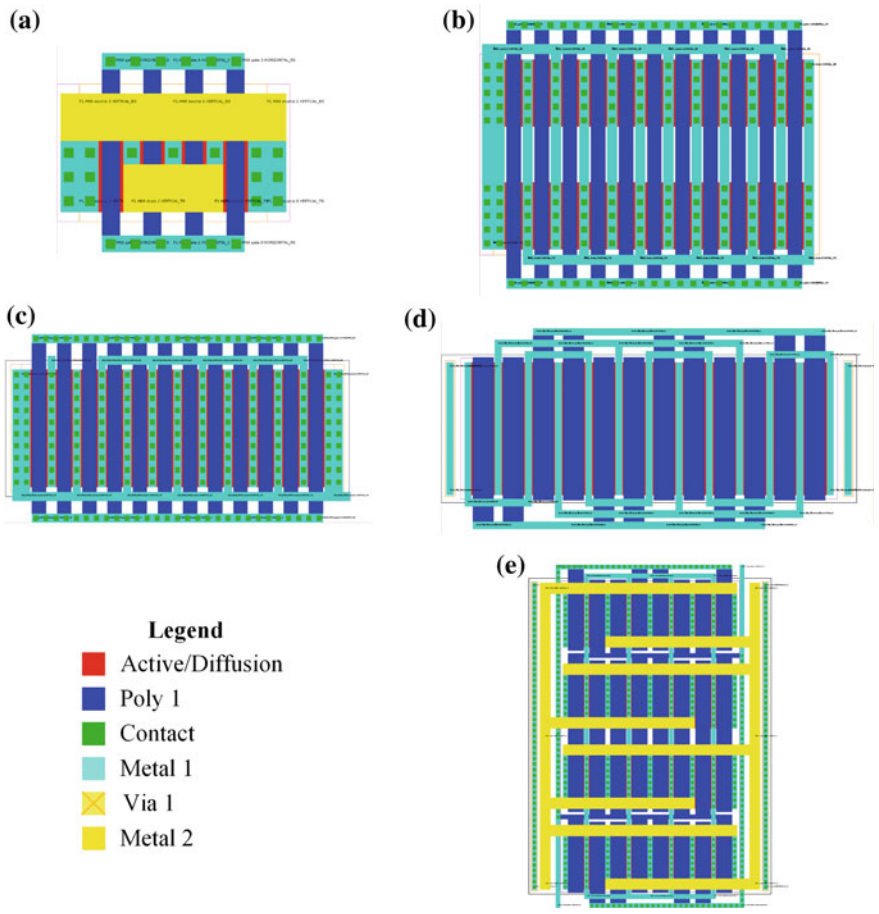


Fig. 6.6 Example of transistors produced by the AMG: **a** Folded transistor of *4 finger*s with connections over the device; **b** Folded transistor with *22 fingers, 2 rows* and connections outside the device; **c** Merge of 2 transistors, one with *2 fingers*, the other with 10, with both gates and sources connected; **d** Interdigitated of two transistors with *6 fingers*; **e** Common-centroid of 2 transistors with *16 fingers* each

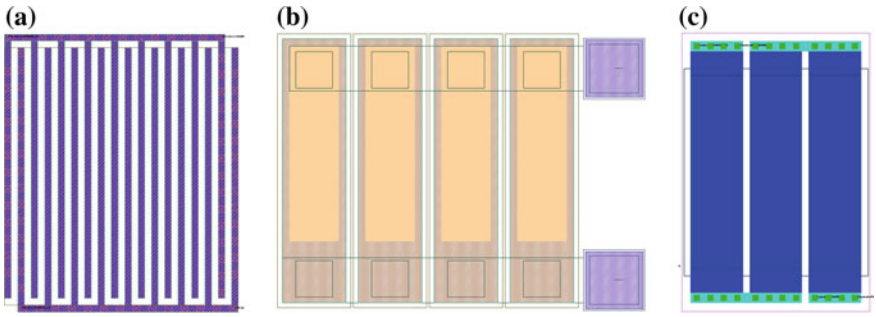


Fig. 6.7 Example of passive devices produced by the AMG. **a** MOM capacitor. **b** MIM capacitor. **c** Polysilicon resistor

all fingers are located in the same stack of active area, the additional device parameter *nrows* defines the number of stacked rows to be drawn. By using multiple rows, large devices are split into smaller ones in different rows. This adds more control over the device aspect ratio, as shown in Fig. 6.6b, while maintaining the original W, L, and number of fingers. Figure 6.6c shows a merged transistor stack, which is created by the abutment of a terminal of two or more transistors. Interdigitated and common centroid transistors-pairs are shown in Fig. 6.6d, e respectively. The first improves matching by interleaving the fingers of two transistors, while the second does the same by arranging the devices in such a manner that they share a common center of mass. The passive devices supported are Metal-Oxide-Metal (MOM) capacitors and polysilicon resistors, and are represented on Fig. 6.7.

Two intra-device transistor routing processes are available, the first, shown in Fig. 6.6a, connects the common terminals over the device using the metal-2 layer, the second, Fig. 6.6b–d, connects the common terminals outside the device using the metal-1 layer. Also, each of the available layout styles provides additional options for bulk/substrate biasing, with different routing solutions. In the AMG the devices are already created in symmetric structures (including intra-cell routing) that facilitate a bottom-up approach in the layout design flow, resulting in a placement that is globally and locally symmetric. Additionally, the device-pair structures that increase matching like common-centroid and interdigitated layout styles reduce the gap between pre- and post-layout simulation. Other device structures, like the technique proposed in [10] to reduce the pre- to post-layout STI stress effect related mismatch could be included in the AMG and corresponding simulation model [13], to further increase the robustness of the solution.

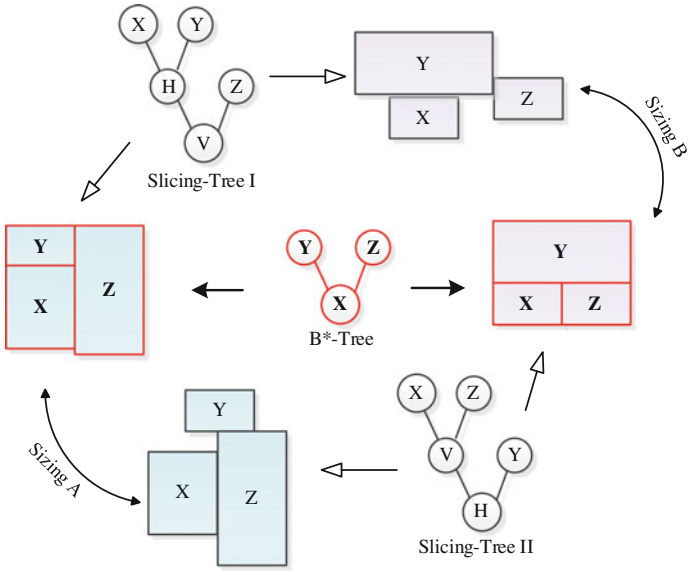


Fig. 6.8 B*-Tree and Slicing-Tree showing the flexibility of the non-slicing structure for different sizes

6.2.3 *Floorplanner*

AIDA-L's floorplanner uses the B*-Tree [14] layout representation to represent the topological constraints. The B*-Tree is used because its' packing yields a non-slicing floorplan. The non-slicing floorplan is less restrictive than the slicing floorplan, and while this does not implies that slicing structures cannot yield good placements, the usage of non-slicing structure leads to more flexibility at the same floorplan definition cost. Figure 6.8 illustrates the flexibility of the non-slicing floorplan when compared to its slicing counterpart. A single B*-Tree generates several slicing placements depending of the sizes of the devices. The slicing tree is used here to represent the slicing floorplan because it is the most common slicing structure. However, and to be fair, the slicing tree can generate non-slicing placements if proceeded by compaction [15].

Figure 6.9 shows some of the B*-Trees that are obtained from three different floorplan for the two stage amplifier introduced in Chap. 3. When the guidelines are loaded, the floorplanner extracts the topological guides into a set of B*-Trees using Algorithm 6.1 from [16]. The sizes in the provided floorplans have no meaning, only the relative positioning regarding the other cells is of concern. The alignment and the topological constraints are inferred from the floorplan placement directly. Beside the B*-trees from the multiple floorplans, the extraction algorithm is not a one to one map, but multiple B*-trees for each floorplan are obtained, which adds another level of flexibility. Therefore, multiple B*-Trees holding the topological relations defined in

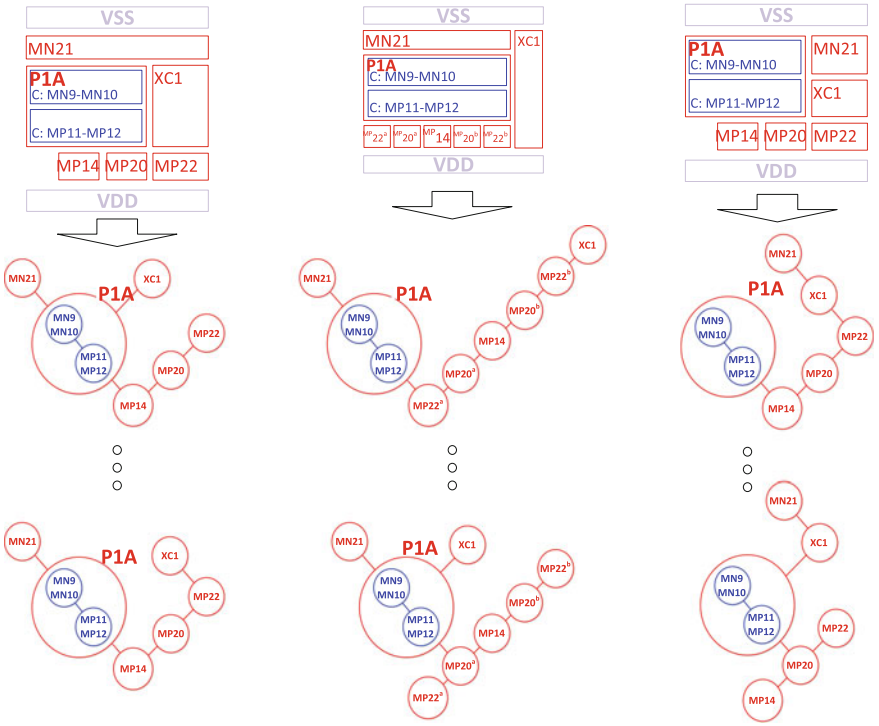


Fig. 6.9 Multiple B*-Trees extracted from the floorplans

the diverse layout guides are used in the evaluation of the geometric properties of all tentative solutions during the optimization loop.

First, the modules are generated with the sizes of the solution being evaluated. During packing the complete device is not really needed, as the floorplanner only needs to know the space that the device occupies. Hence, a simplified method that generates only the bounding for the devices is provided by the AMG, as illustrated in Fig. 6.10. By using this method instead of the complete detailed device generation, the optimization process is further accelerated.

To generate the placement for each of the B*-Trees, the sub-floorplans are packed recursively, and finally, the main B*-Tree is packed. The B*-tree packing is

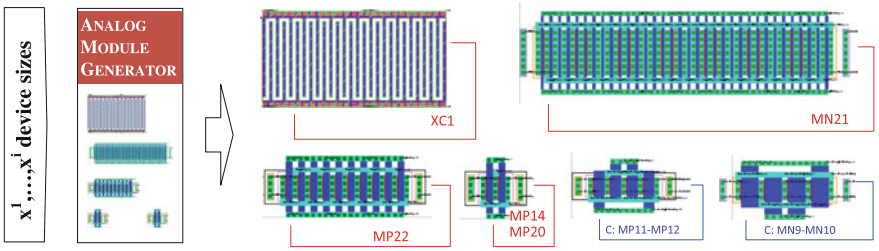


Fig. 6.10 Devices generated using the AMG and corresponding bounding box

performed using the Red-Black interval tree algorithm described in [17]. The complete procedure to evaluate the geometric properties of a tentative solution is detailed in Algorithm 6.1.

Algorithm 6.1 Multi-floorplan estimation procedure

```
    input: List<Floorplan> F (Floorplans with B*-Trees extracted),
    List<Constraint> C (the constraints over floorplan measures,
    e.g., width < 20µm);
    Objective o (the floorplan objetive, e.g., area),
    Solution s (the tentative solution being evaluated)
    output: Floorplan bestF (the floorplan yielding the best objective while
    satisfying the constraints;
    Measures m (the measures associated with the best floorplan)
 1. bestF := null; m := null
 2. for each Floorplan f in F do
 3.     Try
 4.         placement := pack(f, s)
 5.         meas := floorplan.meas //area, width, length, ratio …
 6.     catch AMGCannotInstantiateModuleException
 7.         meas := INVALID_MEAS
 8.     if ((bestF is null) or (isBetter(meas, m, C, o))) then
 9.         bestF:= f; m := meas
10. return (bestF,m)

11. function pack(Floorplan f, Solution s)
12.     lib := {} //empty map
13.     sub-placement:= [] //empty list
        // generate the sub floorplan recursively
14.     for each Floorplan sf in T.sub-floorplans do
15.         placement := pack(sf, s)
16.         sub-placement += placement
17.         lib += {st.id: placement.bounding-box}
        // instantiate the modules (bounding-box only), if the
        // combination of parameters is invalid the Analog Module
        // Generator raises the AMGCannotInstantiateModuleException
18.     for each Module m in T.modules do
19.         bbox := AMG.instantiateBox(m, s)
20.         lib += {m.id: bbox}
        // pack the B*-Tree using the Red-Black interval tree algorithm [143].
        // for the boxes in the lib, it returns the bounding box of the layout
        // and a map with the location of each module/sub-floorplan
21.     (bounding-box, device-placement) := rbBTreePack(t.b-tree, lib)
        // compute the measures associated with the resultant packing
22.     meas := computeMeasures(bounding-box, lib, sub-placement)
23.     return (meas, bounding-box)
24. end function
```

All floorplans are considered and used to produce a placement. After the packing of the floorplan the measures of the best placement are the ones used in the optimizer. The best placement is selected as the one with the best objective, from those

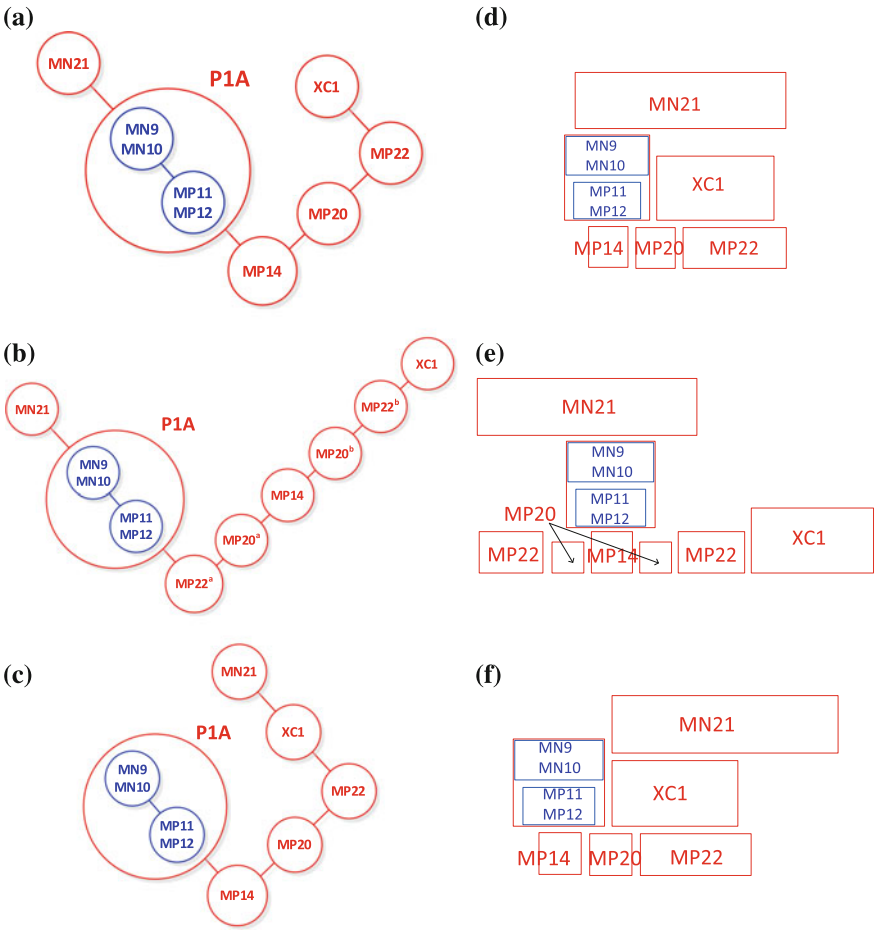


Fig. 6.11 Multiple floorplan packing for a sizing solution: **a**, **b**, **c** B*-Trees; **d** placement for the B*-Tree (**a**) with an area of 361.7 μm^2; **e** placement for the B*-Tree (**b**) with an area of 634.7 μm^2; **f** placement for the B*-Tree (**c**) with an area of 448.6 μm^2

who are feasible, i.e., meet all constraints. If no feasible solutions are found, the most feasible is selected, i.e., the one where the sum of violated constraints is larger (closer to zero). If two solutions are equally infeasible, the one with best objective value is selected. Figure 6.11 illustrates the placement generation using multiple B*-Trees. It shows the placement of a sizing solution obtained from three of the B*-Trees illustrated in Fig. 6.9 using the devices from Fig. 6.10.

Figure 6.12 illustrates the placement shown Fig. 6.11d, which is the best placement in this case, showing the complete device' layouts as obtained form the AMG.

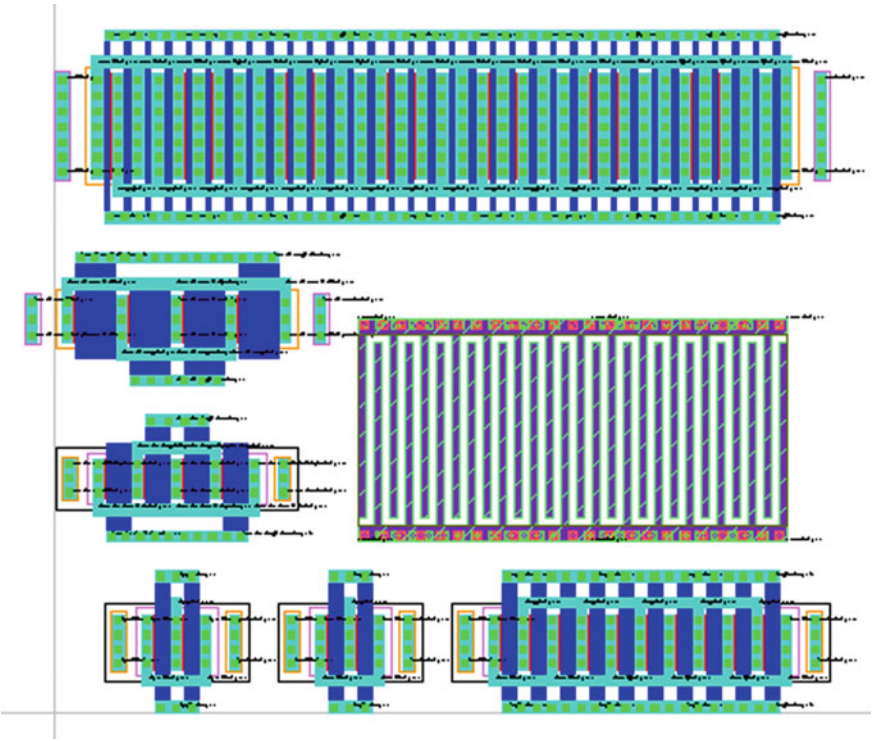


Fig. 6.12 Best placement showing the devices' layout. Reprinted from Integration, the VLSI Journal, 48, Nuno Lourenço, António Canelas, Ricardo Póvoa, Ricardo Martins, Nuno Horta, Floorplanaware analog IC sizing and optimization based on topological constraints, 183–197, Copyright (2015), with permission from Elsevier

The geometric measures obtained by the floorplanner, which can be selected as objectives or constraints, are defined in Eq. (6.1). The *global-width*, *global-height*, *area* and *aspect-ratio* refer to the complete placement, the *empty-area* is the unused space inside the rectangle delimitating the floorplan, the *empty-area-ratio* is the ratio between the empty-area and the area, and the *max-module-ratio* measures the maximum aspect ratio of the modules and is usually used to constrain the shape of the modules, usually used to keep individual devices approximately square.

$$\begin{aligned}
&global-width, global-heigth\\
&area = gloabl-width \times global-height\\
&aspect\text{-ratio} = \tfrac{\text{global-width}}{\text{global-height}}\\
&empty\text{-area} = \text{area} - \textstyle\sum \text{module-area}\\
&empty\text{-area-ratio} = \tfrac{\text{empty-area}}{\text{area}}\\
&max\text{-module-ratio} = \textstyle\max_{i=1\ldots D}(\text{module-ratio}_i)\\
&where\ module\text{-ratio} = \tfrac{\max(\text{module-width},\text{module-height})}{\min(\text{module-width},\text{module-height})}
\end{aligned} \tag{6.1}$$

In this floorplan-aware approach, the parasitic mismatch errors and thermal gradient effects are alleviated using the designer own experience to define where to take special care in the layout synthesis. Although less accurate than works where complete layout and extraction are done, this approach is computationally efficient and less time consuming. Therefore speeding the global sizing optimization while enforcing matching, symmetry and proximity constraints, and evaluating realistic geometrical properties.

6.3 Layout-Aware Circuit Sizing

While the floorplan-aware accounts for geometric properties explicitly, the parasitic effects are handled implicitly by a careful layout and aiming at simulating using models as accurate as possible with respect to layout dependent effects (LDE). The advantage of the floorplan-aware is the reduced impact in the execution time, but for hard specifications the impact of the LDE can still lead to wrongly sized circuits. For the cases where parasitic effects cannot be alleviated by imposing such design constraints, the solution obtained using AIDA-C's floorplan-aware flow can be used as a robust starting point for the layout-aware flow, where the complete layout is generated (placement and routing) and a detailed extraction is done.

The layout generation for accurate parasitic extraction is the bottleneck of layout-aware methodologies. Some approaches rely on parametric cell generators, which lack on flexibility and require long/complex setup before each design. Some approaches avoid parametric generators using full custom generators but they show extremely high execution time. In addition, even these kind of approaches do not automatically update the routing topology to properly fit the multitude of devices' sizes/shapes and performances that are explored on the search for the final solution.

Given the relevance of LDEs in the degradation of the performance, AIDA's flow evolved to cover layout-aware sizing and is illustrated in Fig. 6.13. The tool

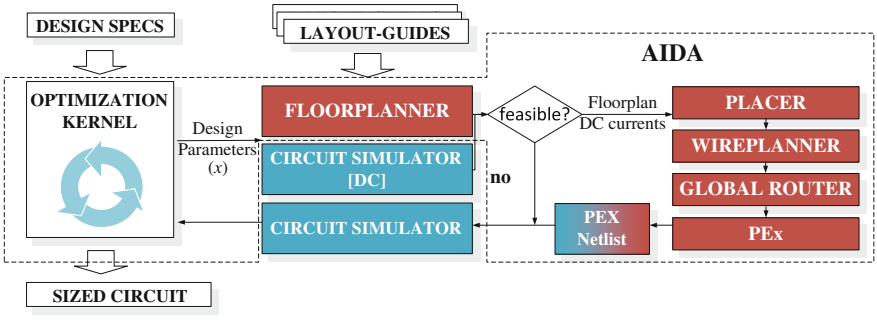


Fig. 6.13 Layout-aware evaluation flow

was updated to include a methodology for automatic layout-aware synthesis of analog ICs that uses a built-in extractor to accurately compute the impact of parasitic from both floorplan and early-stages of routing.

The AIDA–L's wiring planner and global router [18] were integrated in the loop of the automatic sizing, and generate the layout for each tentative solution at each generation. Its inputs are the multiple floorplan constraints (common to all sizing solutions) and the electric-currents of each device terminal (specific for each sizing solution, and obtained with AIDA-C from the DC simulation). The use of DC currents greatly simplifies the automation procedure and provides an approach easy to apply to most circuits, nevertheless, in general, other current values, measured using the proper testbenches, could be used (there is also the cost in execution time, as DC simulation is very time efficient).

Following the loop of the optimization process, the major steps are highlighted by order:

1. AIDA-C optimizer sets x, the design variables, (e.g., devices' widths, lengths, number of fingers, etc.) for the tentative sizing solution being evaluated;
2. The DC constraints (overdrives, saturation margin, etc.) are measured using the electrical simulator, also, the DC electric-currents for each device terminal are obtained. If a solution is unfeasible (i.e., a DC constraint violated) the layout-related data is not considered for further evaluation (still the solution is evaluated in the full set of testbenches; this evaluation is needed for guidance of the optimizer, especially when all tentative solutions fail DC constraints), otherwise, the solution is sent to AIDA-L;
3. AIDA-L generates multiple floorplans, and the devices are placed according to the one that better suits the geometric requirements. The Router, inputs the placement and devises the solution specific electromigration-aware wiring topology and global routing;
4. The parasitics are extracted using the built-in parasitic extractor module and back annotated in the netlist;
5. The parasitic-aware performances are measured from the complete set of testbenches (DC, AC, TRAN, etc.) using the electrical simulator for all defined PVT corners and used together with the accurate geometrical properties of the circuit, measured by the floorplanner, in the optimization process.

6.3.1 Electromigration-Aware Global Router

The next step of the in-loop layout generation is the routing. Given the multitude of solutions, the routing must be adjusted to each sizing/floorplan without any user

intervention. The pre-parasitic netlist is obviously common to all sizing solutions of a circuit. Given the corresponding placement and the electric-currents flowing through the device's terminals, the current-correct tree that provides the optimal terminal-to-terminal connectivity and flows, and that minimizes the wiring area is then computed for each of the circuit's nets as described in [18]. Here, the flow from current-sources, i.e., terminals with a positive current value, is redirected to the current-sinks, i.e., terminals with a negative current value.

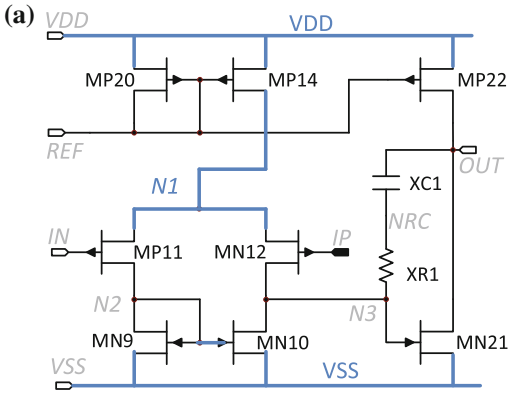
Then, in the global Router, a path-finding algorithm operating over a sparse non-uniform multilayer grid is used to transform the terminal-to-terminal connections into rectilinear paths, where each device terminal defines multiple locations for the connection to be made (multiports). The ports selected from the corresponding terminals' multiports are the ones that minimize the wire area in the absence of obstacles (i.e., design rules are not enforced in the grid). The width of each wire is a function of the electric-current assigned to it during the creation of the wiring topology, subject to electromigration and voltage-drop constraints. Also, wiring symmetry information is automatically extracted and applied in the global paths [19].

There are two important simplifications that are critical for the efficiency of the in-loop routing: first no obstacles are considered in the grid, and second, the grid does not enforce design rules. The resultant paths are then an approximation to the ones that would be obtained after detailed routing, but, as it will be shown in the results, provide close enough parasitic information to be used for sizing optimization. In Fig. 6.14, the electromigration-aware wiring topology and global routing for three different nets (for illustration purposes) are presented.

6.3.2 Parasitic Devices Extraction

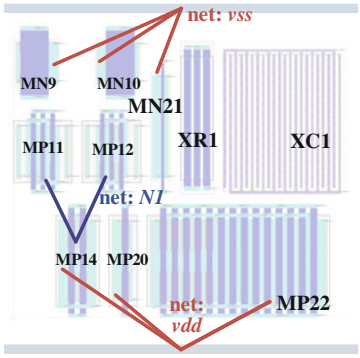
The next source of layout related data to be considered is parasitic devices. By avoiding an external extractor and taking advantage of the correct-by-construction placement and interconnect estimates it is possible to efficiently include parasitic information in the optimization loop. For MOS transistors it is common the use of geometric methods for the intra-device parasitics, where the device's width, length and number of fingers are used in an equation that provides the estimated values. However, the technology-dependency of those equations, the difficulty found to derive them for more complex layout styles, e.g., common centroid, and the inapplicability to wires, pressed for a different approach. Hence the general procedure used is also applied for the intra-device parasitics.

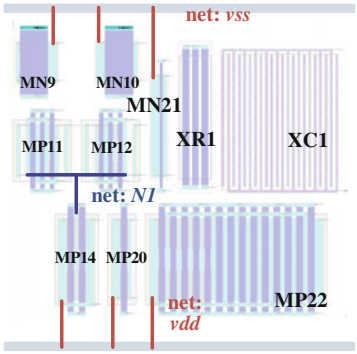
The estimated parasitic resistances of the interconnects, parasitic capacitances between terminals' shapes of the devices, between terminals' shapes and interconnects, and between pairs of global routing interconnects are detailed in this



(b)

vdd	Vss	N1
$MP20_{src}$@10µA	$MN9_{src}$@–9.97µA	$MP14_{dr}$@–19.93µA
$MP14_{src}$@19.93µA	$MN10_{src}$@–9.96µA	$MP11_{src}$@9.97µA
$MP22_{src}$@146.1µA	$MN21_{src}$@–146.1µA	$MN10_{src}$@9.96µA
vdd@–176.03µA	vss@166.03µA	





vdd	Vss	N1
(vdd→$MP20_{src}$)@10µA	(vss→$MN9_{src}$)@9.97µA	($MP14_{dr}$→Steiner)@ 19.93µA
(vdd→$MP14_{src}$)@9.69µA	(vss→ $MN10_{src}$)@9.69µA	(Steiner →$MP10_{src}$)@ 9.97µA
(vdd→$MP22_{src}$)@146.1µA	(vss→$MN21_{src}$)@ 146.1µA	(Steiner→$MP10_{src}$)@ 9.96µA

Fig. 6.14 Global routing procedure: **a** Schematic highlighting the illustrated nets; **b** Netlist and generic electric-currents associated to each terminal; **c** EM-aware wiring topology and **d** global routing. The wires' widths are function of the electric-current imposed on them

subsection. To avoid an external extractor, reliable estimates for the parasitic devices are derived internally using the in-loop extractor that follows the loops outlined in Algorithm 6.1.

Algorithm 6.2 RC extraction procedure

```
input: List<Devices<Terminal<Shapes>>> Floorplan (shapes in the floorplan),
List<List<Edge<w_{i,j}, List<Segments>>>> GlobalRouting (global routing
solution, each Edge represents a terminal-to-terminal connection defined by
a path List<Segments>)
output: Map<TerminalPair<Terminal, Terminal>, Double> termPCap (Parasitic
Capacitances between Terminals)
Map<EdgePair<Edge, Edge >, Double> edgePCap (Parasitic Edge Resistance and
Parasitic Cap. between Edges)
Map<Pair<Terminal, Edge >, Double> term2edgePCap (Parasitic Capacitances
between Terminals and Edges)
Map<Terminal|Edge, Double> subPCap (Parasitic Capacitances to the
Substrate)
// Parasitic Capacitances between Terminals
1. devices := new List<Devices> (Floorplan.devices)
2. termPCap := new Map<TerminalPair<Terminal, Terminal>, Double>
3. foreach Device d1 in devices do
4.     devices:= devices - d1
5.     foreach Terminal t1 in d1 do
6.         pcap = 0
7.         foreach Device d2 in devices do
8.             foreach Terminal t2 in d2 do
9.                 if(t1 is far from t2) continue
10.                foreach Shape sh1 in t1 do
11.                    foreach Shape sh2 in t2 do
12.                        pcap += parasitic capacitance between sh1 and sh1
13.                termPCap.terminalPair(t1,t2).value:=pcap
// Parasitic Edge Resistance and Parasitic Cap. between Edges
14. nets := new List<Net> (GlobalRouting)
15. edgePCap := new Map<EdgePair<Edge, Edge >, Double>
16. foreach Net n1 in GlobalRouting do
17.     nets:= nets -n1
18.     foreach Edge e1 in n1 do
19.        foreach Segment s1 in e1 do
20.            e1.pRes += compute resistance by square counting
21.                  // considers the wire's width w_{i,j} and segment's length
22.            foreach Net n2 in nets do
23.                foreach Edge e2 in n2 do
24.                    foreach Segment s2 in e2 do
25.                        pcap := compute parasitic cap. between s1 and s2
26.                    edgePCap.edgePair(e1,e2).value += pcap
// - Parasitic Capacitances between Terminals and Edges
// - Parasitic Capacitances to the Substrate
//follow the same principles above for terminals or edges processing
```

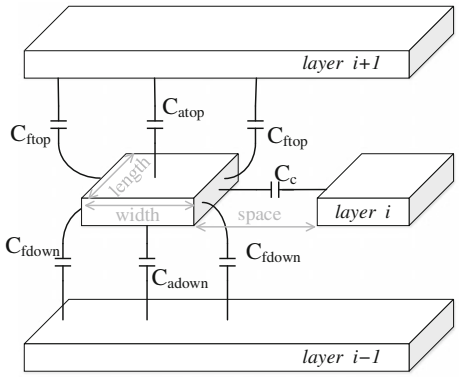


Fig. 6.15 Different capacitances considered in the 2.5-D model

6.3.2.1 Parasitic Capacitances

The unitary parasitic capacitances estimated are: the capacitance between metal stripe and substrate/active area; overlap capacitance between two overlapping metal stripes; and the parallel coupling capacitance between adjacent metal stripes. For capacitive parasitic extraction the alternatives found in the literature are: the computationally expensive numerical methods, which are hardly usable within the optimization loop; approximated analytic methods obtained from electromagnetic models; or empirical techniques, where the extensive experimental data provided by the foundry for a particular technology is used and fitted to a certain model [20].

The extraction is done using an empirical-based 2.5-D geometrical model [21], illustrated in Fig. 6.15, based on the technology-dependent capacitance tables found in the PDKs, where, several realistic 3-D effects, e.g., coupling capacitances between two interconnects, crossings in different conductors, etc., are modeled as a combination of 2-D structures.

In the computation of the parasitics, all interconnects are assumed to run by default on the first metal layer. As a consequence of the simplified routing that is applied, sometimes two metal strips will overlap, in this case, the parasitic capacitance is computed considering one of them to be in the second metal layer instead. In Fig. 6.16 the PMOS device, M1, from the floorplan shown in Fig. 6.16a, is used, from Fig. 6.16b–d, to illustrate the shapes considered to compute the intra device capacitances. In Fig. 6.16e, the parasitic capacitance between terminals of different devices, and, in Fig. 6.16f, the parasitic capacitance between a device terminal and a path of the global routing.

As mentioned, estimation of intra-device terminal to terminal parasitics could be further accelerated by analytic methods, where the device's width, length and number of fingers are used in an equation that provides the estimated. In the implemented approach this acceleration was not considered, so that the parasitic extractor module of the tool is kept separated from the device generator.

The substrate/active area capacitance C_S is the sum of the capacitive component between the shapes of a device's terminal (or interconnect) to the plane below, i.e.,

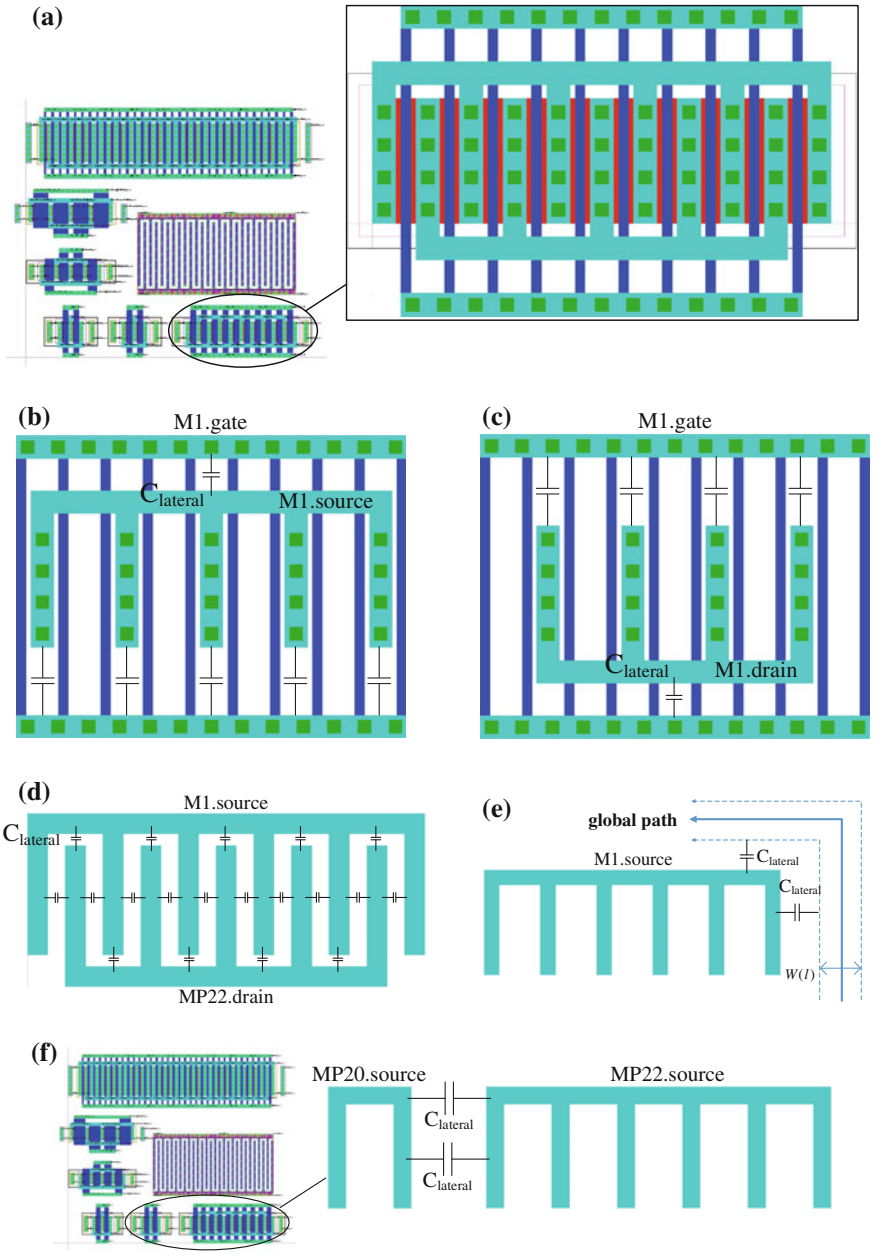


Fig. 6.16 Extracted capacitors and shapes considered: **a** transistor MOSFET M1 zoomed from the floorplan; **b** Transistor's shapes considered to compute the capacitance gate-source C_{gs}; **c** Transistor's shapes considered to compute the capacitance gate-drain C_{gd}; **d** Transistor's shapes considered to compute the capacitance drain-source C_{ds}; Some $C_{lateral}$ components were illustrated, the total parasitic capacitance is the sum of all partial parasitic components (**e**) between a device's terminal and a path of the global routing, and, on (**f**), between terminals of a different devices

substrate, well or active areas, and is computed according to Eq. (6.2), where the area capacitive component C_a and fringe capacitive component C_f are obtained by interpolation of the technology-dependent data provided by the foundry in function of the conductor's *width*.

$$C_S = (C_a(width) \times width) + (2 \times Cf(width) \times length) \tag{6.2}$$

The Coupling Capacitance is computed between two conductors (shapes of a device's terminal or interconnect) on different layers. The coupling capacitance $C_{coupling}$ has also area and fringe components, as defined in Eq. (6.3). If no area component C_a exists the coupling capacitance is ignored.

$$C_{Coupling} = (C_a(width) \times overlapwidth) + (2 \times Cf(width) \times overlaplength) \tag{6.3}$$

The lateral capacitance $C_{lateral}$ is considered between two conductors running in parallel or perpendicular on the same layer. Here, the fringe components are negligible, and is defined as: where $C_c(width, space)$ is the coupling capacitive component obtained by the interpolation in conductor's *width* and *space* of the foundry provided values. The C_c value for distant parallel or narrow perpendicular conductors tends to zero.

$$C_{lateral} = C_c(width, space) \times parallel\_length \tag{6.4}$$

6.3.2.2 Parasitic Resistance

The parasitic resistance $R_{parasitic}$ for each wire's segment, which is defined by a width and length, is computed from the resistance per unit quare R_c tabulated for a given conductor at a temperature T:

$$R_{parasitic} = R_C(T) \times \frac{length}{width} \tag{6.5}$$

Each wire has any number of segments/bends, so all partial resistance components are considered for square counting, as illustrated on Fig. 6.17.

6.3.3 *Back-Annotation and Simulation of the Parasitics*

With the new wire topology and the values for the corresponding parasitic devices, the nets in the original netlist are updated to account for the parasitic effects, where each wire is modeled as a lumped transmission line using the $\pi 2$ model [22], as

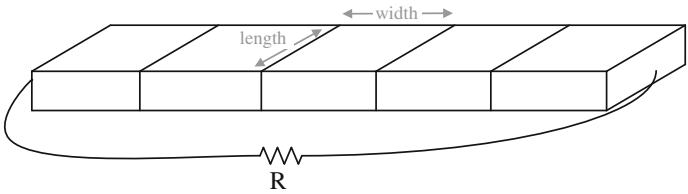


Fig. 6.17 Parasitic interconnect resistance computed by square counting

Fig. 6.18 $\pi 2$ model for the wires

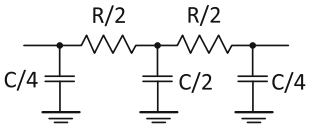
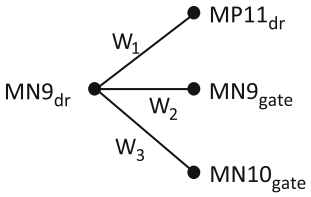


Fig. 6.19 Wiring topology for net N2



shown in Fig. 6.18. In addition, the parasitic capacities between power supply nets, e.g., *vdd*, *vss*, that bias the plane below terminals and wires are added to their bulk capacity.

The net "N2" from Fig. 6.14 is used to illustrate the procedure for creating the parasitic netlist in Fig. 6.20. The electromigration-aware wire topology, shown in Fig. 6.19, is used to replace the net by a set of the individual wires, resulting in the sub-circuit shown in Fig. 6.20a. The terminal nodes are also loaded with the coupling the capacity between the terminal metallization and bulk, illustrated in Fig. 6.20b.

The previous flow creates parallel capacitors. Since the netlist is used inside the optimization loop, the redundant devices are reduced to speed the pre-processing of the simulator. Instead of the multiple capacitors, a single capacitor with the total capacitance is created instead leading to the final sub-circuit for CR capacities shown in Fig. 6.20c.

The coupling capacities are then added between the corresponding nodes. Figure 6.21 shows the several coupling capacitances considered. The coupling capacitance between terminals is added between the corresponding nodes, the coupling capacitor between wires is connected between the central node of the wires [23], and finally, the coupling between wires and terminals is connected between the wire central node and the terminal node.

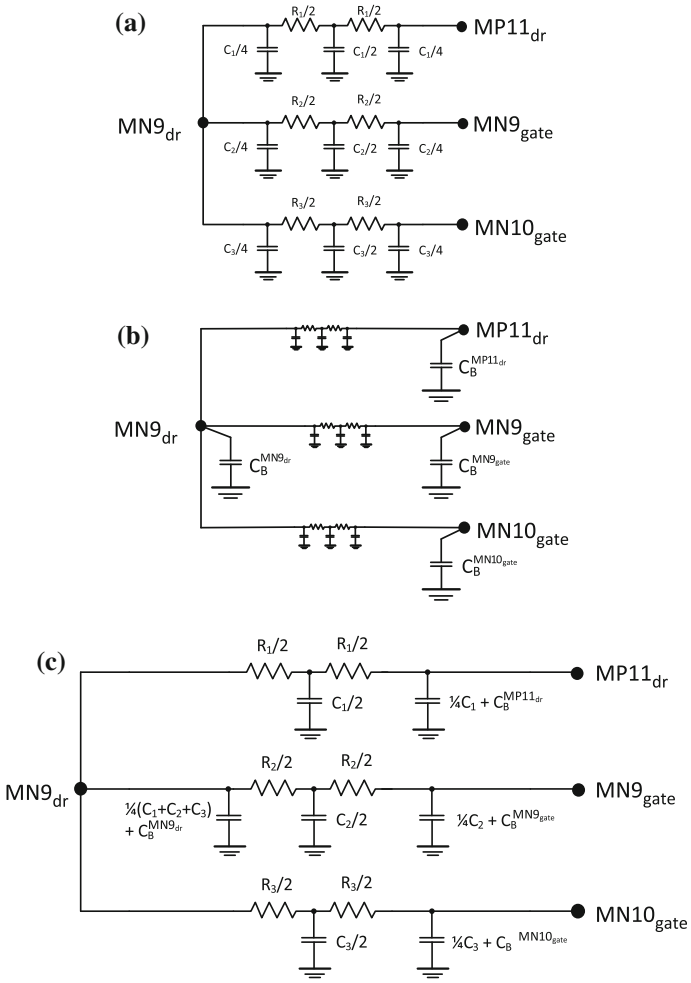


Fig. 6.20 Parasitic netlist for net N2: **a** N2: wires parasitic RC devices; **b** N2: terminal-bulk capacitors; **c** N2: resistors and bulk capacitors

In Table 6.1 some post-layout simulation results for the single ended 2-stage amplifier using the developed AIDA's parasitic extractor module and a commercial tool are presented. The parasitic effects are progressively accounted for, showing their impact in the circuit performance. First only local coupling parasitics between terminals in the same device are accounted. This coupling appears due to the intra-device routing. As mentioned previously, these parasitics are passible of being modeled analytically as a function of the device parameters to reduce the gap between the pre- and post-layout simulation, but each time the AMG is updated the parasitic expressions need to be updated. In the implemented flow they were also extracted to avoid the need for these updates. Secondly, the coupling between

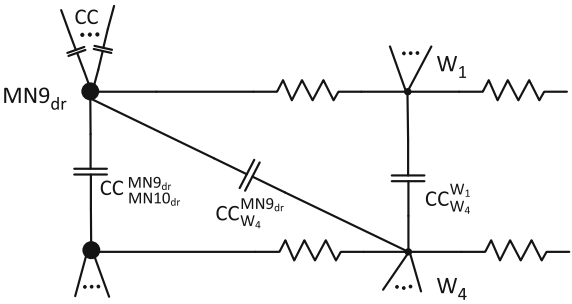


Fig. 6.21 Parasitic netlist coupling capacitances: terminal-terminal, terminal-wire and wire-wire

Table 6.1 Post-layout measures: partial and complete extraction with AIDA and with Mentor Graphics' Calibre®

	MEASURE	NETLIST[1]	Implemented parasitic extractor			Calibre®[5]
			I^2	I + T[3]	I + T + W[4]	
D1	GBW (MHz)	55.9	54.3	53.8	53.5	53.0
	PM (°)	54.6	54.3	54.3	53.9	53.7
	NO (μVrms)	180.4	178.5	177.9	178.1	177.6
D2	GBW (MHz)	88.2	84.9	84.2	83.7	82.7
	PM (°)	53.3	53.1	52.8	52.1	52.2
	NO (μVrms)	203.7	200.4	200.2	200.9	199.7

[1]In AIDA parasitic are extracted without generating the detailed layout
[2]I—Intra-device coupling parasitics, i.e., only the coupling between module terminals are considered (i.e., C_{gs}, C_{gd}, C_{ds}, C_{gb}, C_{db}, C_{sb})
[3]I + T—The parasitic in I^2 together with coupling parasitics between terminals of all modules in the floorplan
[4]I + T + W—The parasitics for the interconnects are considered in addition to the previous ones
[5]Extracted with Mentor Graphics' Calibre®, detailed routing generated with AIDA

adjacent devices in the floorplan is also accounted, finally the coupling between the wires and between wires and terminals are also considered.

As the values show, intra-device parasitics, commonly unnecessarily overlooked in the pre-layout simulation, despite being easy to include accurately in the models by fixing the device-level layout design (as done in RF models), can have a significant impact in the circuit performance. However for the accurate evaluation of the circuits' performance in the presence of parasitics the wires contributions are indispensable. The precision lost when using the lightweight AIDA's parasitic extraction when comparing to Calibre®, is not only rewarded by the small extraction time required for this simplified parasitic extractor, but specially by the fact that the built-in procedure operates over a global routing avoiding the need of the final detailed layout during the optimization iterations. Still it confirms the precision is sufficient to guide the parasitic-aware optimization in the right direction.

6.4 Conclusions

In this chapter, the inclusion of layout effects in AIDA's sizing optimization is presented. The floorplan-aware flow is based on relaxation, where only geometric information is considered explicitly. While less accurate then methods with complete layout generation and extraction, the floorplan-aware is extremely efficient, allowing the simultaneous global optimization of both device sizes and circuit layout. The common pitfalls that arise from using templates, complicated setup and low reusability with different specifications are here handled in a twofold, first, floorplan description is kept as simple as possible, also generated semi-automatically from the netlist with only a small set of parameters left for being defined, and second, by using multiple floorplans the design handles new specifications better. Then, an innovative layout-aware evaluation where the inclusion of parasitic devices is done using the floorplan and global routing is presented. This approach does not require any extra setup, and the routing approximation obtained from the global router makes the evaluation fast while increasing the accuracy. The applicability of the approach is demonstrated in the next chapter of this book with post-layout performances attained for the circuits designed with the AIDA-C layout-aware methodology, versus, the traditional optimization-based sizing.

References

1. Gielen GGE (2007) Design tool solutions for mixed-signal/RF circuit design in CMOS nanometer technologies. In: Asia and South Pacific design automation conference (ASP-DAC'07), 2007
2. Castro-Lopez R, Guerra O, Roca E, Fernandez FV (2008) An integrated layout-synthesis approach for analog ICs. IEEE Trans Comput Aided Des Integr Circuits Syst 27(7): 1179–1189
3. Pradhan A, Vemuri R (2009) Efficient synthesis of a uniformly spread layout aware Pareto surface for analog circuits. In: International conference on VLSI design, New Delhi, 2009
4. Ranjan M et al (2004) Fast, layout-inclusive analog circuit synthesis using pre-compiled parasitic-aware symbolic performance models. In: Design, automation and test in europe conference and exhibition, Paris, 2004
5. Habal H, Graeb H (2011) Constraint-based layout-driven sizing of analog circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 30(8):1089–1102
6. Lourenco N, Martins R, Horta N (2015) Layout-aware sizing of analog ICs using floorplan and routing estimates for parasitic extraction. In: 2015 Design, automation and test in Europe conference and exhibition (DATE), Grenoble, 9–13 March 2015
7. Martins R, Lourenco N, Canelas A, Povoa R, Horta N (2015) AIDA: Robust layout-aware synthesis of analog ICs including sizing and layout generation. In: International Conference on synthesis, modeling, analysis and simulation methods and applications to circuit design (SMACD), Istanbul, 7–9 Sept 2015
8. Hastings A (2005) The art of analog layout, 2nd edn. Prentice Hall, Upper Saddle River
9. Pelgrom MJM, Duinmaijer Aad C J, Welbers APG (1989) Matching properties of MOS transistors. IEEE J Solid-State Circuits 24(5):1433–1439

10. Lewyn L, Williams N (2011) Is a new paradigm for nanoscale analog cmos design needed? Proc IEEE 99(1):3–6
11. Canelas A, Martins R, Póvoa R, Lourenço N, Guilherme J, Horta N (2015) Enhancing an automatic analog IC design flow by using a technology-independent module generator, In: Fakhfakh M, Tlelo-Cuautle E, Fino M (eds) Performance Optimization techniques in analog, mixed-signal, and radio-frequency circuit design. IGI Global, Hershey PA, pp 102–133
12. Yilmaz E, Dundar G (2009) Analog layout generator for CMOS circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 28(1):32–45
13. Paydavosi N et al (2016) BSIM4v4.8.0 MOSFET Model manual. http://www-device.eecs.berkeley.edu/bsim/Files/BSIM4/BSIM480/BSIM480_Manual.pdf. Accessed 12 April 2016
14. Chang Y-C, Chang Y-W, Wu G-M, Wu S-W (2000) B*-trees: a new representation for non-slicing floorplans. In: Proceedings of design automation conference 2000, Los Angeles, 2000
15. Minghorng L Wong DG (2001) Slicing tree is a complete floorplan representation. In: Design, automation and test in Europe conference and exhibition (DATE), Munich, 13–16 March 2001
16. Martins R, Lourenço N, Horta N (2013) LAYGEN II—Automatic layout generation of analog integrated circuits. IEEE Trans Comput Aided Des 32(11):1641–1654
17. Balasa F, Maruvada SC, Krishnamoorthy K (2003) Using red-black interval trees in device-level analog placement with symmetry constraints. In: Asia and South Pacific design automation conference, Kitakyushu, 2003
18. Martins R, Lourenço N, Canelas A, Horta N (2014) Electromigration-aware analog router with multilayer multiport terminal structures. Integr VLSI J 47(4):532–547
19. Martins R, Lourenço N, Canelas A, Horta N (2015) Extraction and application of wiring symmetry rules to route analog multiport terminals. In: IEEE international symposium on circuits and systems (ISCAS), Lisbon, 2015
20. Shomalnasab G, Heys H, Zhang L (2013) Analytic modeling of interconnect capacitance in submicron and nanometer technologies. In: IEEE International symposium on circuits and systems, 2013
21. Kao W, Lo C-Y, Basel M, Singh R (2001) Parasitic extraction current state of the art and future trends. Proc IEEE 89(5):729–739
22. Gong J, Pan DZ, Srinivas PV (2001) Improved crosstalk modeling for noise constrained interconnect optimization. In: Asia and South Pacific design automation conference, 2001
23. Bai X, Chandra R, Dey S, Srinivas PV (2003) Noise-aware driver modeling for nanometer technology. In: International symposium on quality electronic design, 2003

Chapter 7
AIDA-C Layout-Aware Circuit Sizing Results

7.1 Single Stage Folded Cascode Amplifier with Bias

The folded cascode amplifier introduced in Chap. 5 of this book did not included the biasing circuitry. In this test case the circuit was extended to include the biasing circuitry, the amplifier and bias schematics are shown in Fig. 7.1. In addition, the layout guides, illustrated in Fig. 7.2, were also defined. A new set of ranges, objectives and specifications is listed in Tables 7.1 and 7.2. The problem has 25 optimization variables, 2 objectives and 20 constraints, and also, 8 corner cases are considered: (1) Bias voltage is 10 % above the nominal value and temperature is 55; (2) Bias voltage is 10 % above the nominal value and temperature is 0; (3) Bias voltage is 10 % below the nominal value and temperature is 55; (4) Bias voltage is 10 % below the nominal value and temperature is 0; (5) Slow NMOS and slow PMOS; (6) Slow NMOS and fast PMOS; (7) Fast NMOS and slow PMOS; and (8) Fast NMOS and Fast PMOS.

The circuit was optimized using the typical followed by corners (TC) strategy described in Chap. 4 of this book, with a population of 256 and 800 generations for the initial typical optimization and, 128 elements and 400 generations for the later corners optimization. Figure 7.3 shows the Pareto fronts for both typical and corners stages of the synthesis. Figure 7.4 shows the floorplans for the 4 points marked in the Pareto optimal front (POF).

By taking into account the impact of the sizing in the circuit geometry, AIDA-C [1–4] can use electrically equivalent, but geometrically different versions of the same devices to obtain a packed layout. If the layout is not taken into account, manual definition of the number of fingers that yield a better layout must be done, this process is long and iterative, and changing the number of fingers may impact the circuit performance requiring further simulations to ensure that all specifications are still meet.

Moreover, multiple solutions all meeting the performance and geometric specifications are presented together with their respective floorplan, for an easy and fast

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0_7

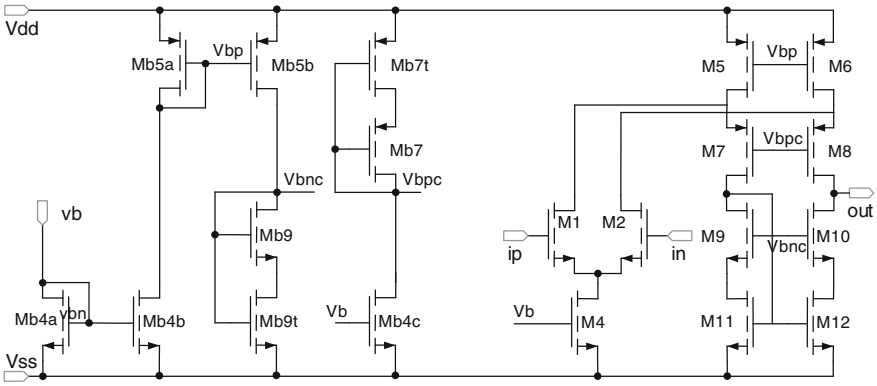


Fig. 7.1 Schematic of the single-ended folded cascode amplifier with bias

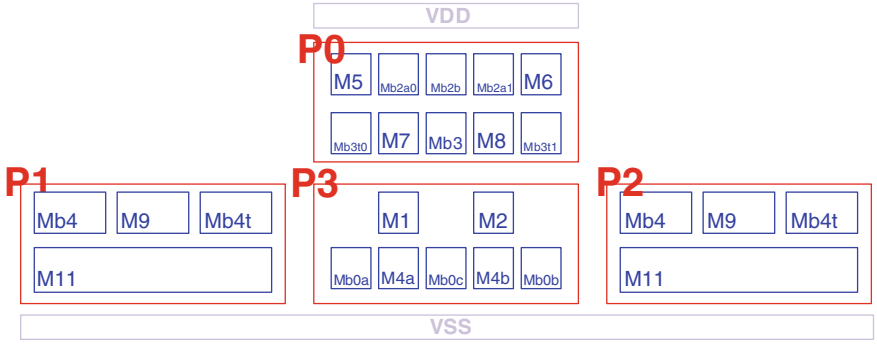


Fig. 7.2 Floorplan template for the single-ended folded cascode amplifier with bias

Table 7.1 Variables and ranges for the single-ended folded cascode amplifier with bias

Variable (unit)	Min.	Grid resolution	Max.
l1, l4, l5, l7, lb7, l9, lb9, l11 (nm)	120	10	10,000
w1, w4, w5, w7, wb7, w9, wb9, w11 (μm)	0.24	0.1	400.0
nf1, nf4, nf5, nf7, nfb7nf9, nfb9, nf11	2	2	50
Ib (μA)	1.0	1	4000

The variables l1, w1 and nf1 are the dimensions and number of fingers of M1 and M2; l4 w4, and nf4 of M4; l5, w5 and nf5 of M5 and M6; l7, w7 and nf7 of M7 and M8; l9 w9, and nf9 of M9 and M10; l11, w11 and nf11 of M11 and M12, lb7/9, wb7/9 and nfb7/9 are the dimensions and number of fingers of the unmatched devices Mb7/9t, the other bias devices are matched in a 4:1 relation with the main amplifier

Table 7.2 Objectives and specifications for the single-ended folded cascode amplifier with bias

Specifications	Measure	Target	Description
Objectives	Area (μm)	Minimize	Area
	a0 (dB)	Maximize	DC Gain
Constraints	gbw (MHz)	≥ 50	Unit-gain frequency
	a0 (dB)	≥ 40	DC Gain
	pm (°)	$60 \leq$ pm ≤ 90	Phase margin
	ov[a] (mV)	≥ 50	Overdrive voltages ($V_{GS} - V_{TH}$)[b]
	d[a] (V/V)	≥ 1.15	Saturation Ratio ($V_{DS} - VD_{Sat}$)/VD^{b}_{Sat}
	ar (m/m)	$0.5 \leq$ ar ≤ 2	Aspect ratio
	elr (m/m)	$0.33 \leq$ elr ≤ 3	Individual device aspect ratio

[a]The constraint applies to: M1, M4, M5, M7, M9 and M11
[b]For PMOS devices the overdrive is $V_t - V_{gs}$ and delta is $V_{dsat} - V_{ds}$

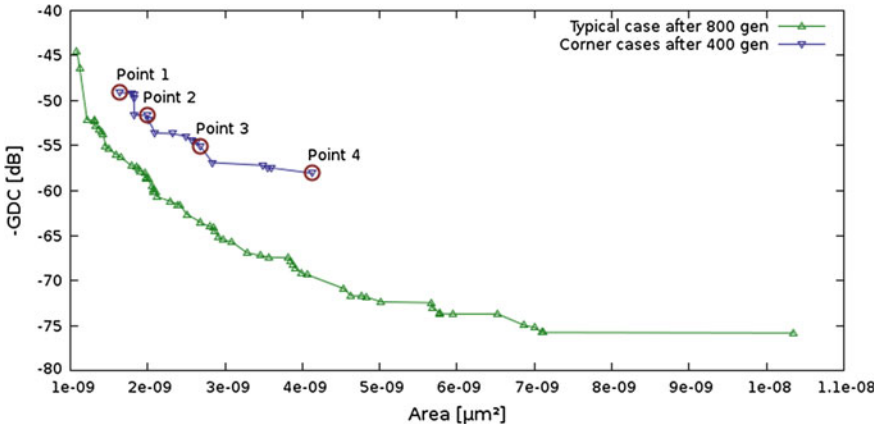


Fig. 7.3 POF obtained with floorplan-aware sizing of the folded cascode amplifier with bias

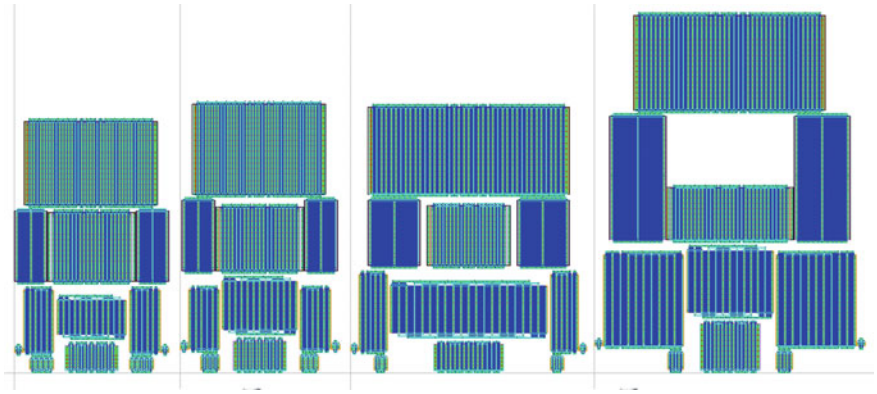


Fig. 7.4 Floorplan of the 4 designs for the folded cascode amplifier with bias marked in the corners' POF: Point 1 to Point 4. All floorplans placed at the same scale

selection by the designer. It is also seen that by using only one set of floorplan guides for the entire span of solutions starts do degrade the compactness of the available layout. In Sect. 7.2.1 a complete study on the usage of multiple floorplan guides is provided, showing the advantage in their use.

7.2 Two-Stage Miller Amplifier

In this test-case the two-stage Miller amplifier shown in Fig. 7.5 is used to experiment with multiple floorplan templates [3], a revised version of the circuit is used for layout-aware optimization.

7.2.1 Floorplan-Aware Design

For the floorplan-aware design, the circuit loaded with a 10 MΩ resistor in parallel with 1 pF capacitor and, biased with a current of 10 μA is here optimized for two objectives: maximize the DC gain while minimizing the circuit area. The design variables are the width, length, number of fingers and the number of rows of the MOS devices, and the length and number of fingers of the MOM capacitor. The variable ranges are indicated in Table 7.3. Table 7.4 indicates the design specifications for this circuit working as a versatile low power DC buffer.

As seen in the previous example, a single floorplan is limitative for the entire set of solutions presented by a multi-objective optimization-based sizing approach such as AIDA. In this case study multiple floorplans are used instead. The layout guides are defined by the 3 floorplans, named T1, T2 and T3, shown in Fig. 7.6a–c, respectively, combined with the 4 sub-floorplans for partition P1 shown in Fig. 7.7a–d.

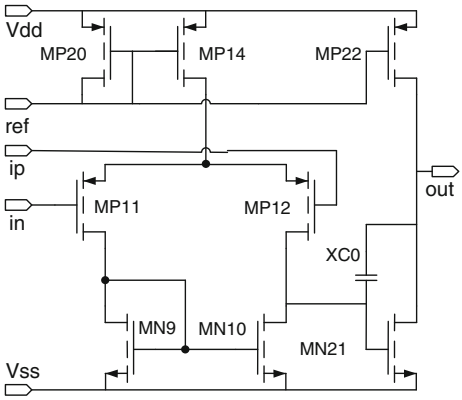


Fig. 7.5 Schematic of the two-stage Miller amplifier

Table 7.3 Variables and ranges for the two-stage Miller amplifier

Variable (unit)	Min.	Resolution	Max.
lc (μm)	4.4	0.1	100
Nfc	14	2	198
lb, lp, lal, l2g (nm)	120	5	1000
wb, wp, wal, w2g (μm)	1	0.1	10
nfbp, nfb2, nfp, nfal, nf2g	1	2	200
nrb, nrp, nral, nr2g	1	1	4

The variables lc and nfc are the length and number of fingers of the MOM capacitor XC0; lb, and wb are the length, and width of MP20; lb, wb, and nfbp are the length, width, and number of fingers of M14; lb, wb, and nfb2 of MP22; lp, wp, and nfp of MP11 and MP12; lal, wal, and nfal of MN9 and MN10; l2g, w2g, and nf2g of MN21; nrb is the number of stacked rows of MP20, MP14 and MP22; nrp the number of stacked rows of MP11 and MP12; nral of MN9 and MN10; and finally nr2g of MN21

Table 7.4 Objectives and specifications for the two-stage Miller amplifier

Specifications	Measure	Target	Description
Objectives	IDD (μA)	Minimize	Current consumption
	GBW (MHz)	Maximize	Unity gain frequency
Constraints	IDD (μA)	≤ 80	Current consumption
	GDC (dB)	≥ 50	Low-frequency gain
	GBW (MHz)	≥ 1	Unity gain frequency
	PM (g°)	≥ 55	Phase margin
	PSRR (dB)	≥ 55	Power supply rejection ratio
	SR (V/μs)	≥ 0.8	slew rate
	Voff (mV)	≤ 1	Offset voltage
	No (μVrms)	≤ 400	Noise RMS
	Sn (nV/$\sqrt{\text{Hz}}$)	≤ 100	Noise density
	OV[1] (mV)	≥ 100	Overdrive voltages ($V_{GS} - V_{TH}$)[b]
	D[1] (mV)	≥ 50	Saturation margin ($V_{DS} - VD_{Sat}$)[b]

[a]The constraint applies to: MP11, MP12, MP14, MP20, MN21 and MP22
[b]For PMOS devices the overdrive is $V_t - V_{gs}$ and delta is $V_{dsat} - V_{ds}$

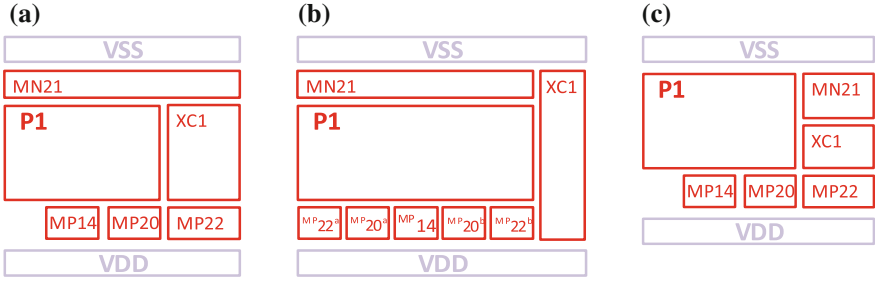


Fig. 7.6 Top-level floorplans for the two-stage Miller: **a** floorplan T1; **b** floorplan T2; **c** floorplan T3

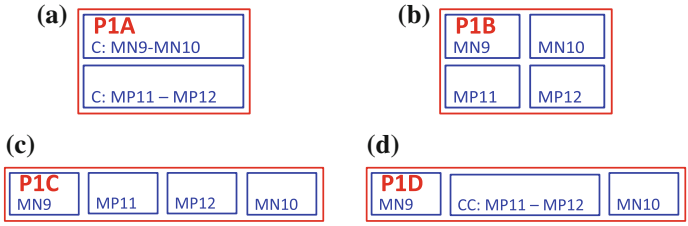


Fig. 7.7 Floorplans for partition P1 of the two-stage Miller amplifier: **a** floorplan P1A; **b** floorplan P1B; **c** floorplan P1C; **d** floorplan P1D. Reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

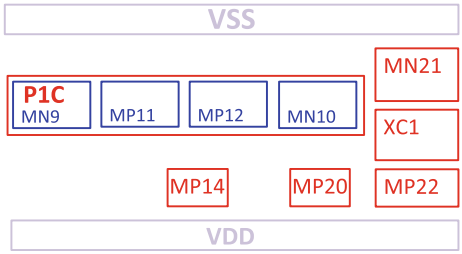


Fig. 7.8 Floorplan T3c for the two-stage Miller amplifier obtained from the combination of T3 with P1c. Reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

Their combination originates 12 (3 × 4) floorplan guides, named by the concatenation of the top-level floorplan name and the version of the partition floorplan. To illustrate the combination process, floorplan T3c, the combination of floorplan T3 with partition P1c, is shown in Fig. 7.8. The floorplan design was implemented in this way to illustrate the advantages of reusing sub-floorplans, thus reducing the workload in the definition of floorplan guides.

To evaluate the impact of the multi-floorplan-aware sizing optimization implemented in AIDA, four scenarios are considered, all with the same parameters and conditions except in the evaluation of the circuit's area. First, the circuit's area is estimated from the sum of the area of the devices (no floorplan is considered). Second, two scenarios where only one floorplan is considered, T1a and T3c respectively, when evaluating the circuit's area, and finally, all twelve floorplan guides are included in the evaluation of the area.

For each scenario, five runs were executed and the results are shown in Fig. 7.9. Figure 7.9 also shown the corresponding worst case Pareto front (WCPF), i.e., the front obtained by merging all Pareto fronts and selecting only those elements that do not dominate any other, which is considered in comparison of the multiple scenarios. Table 7.5 summarizes the results obtained comparing the solutions found in the various scenarios for multiple values of DC gain. On a first note, is relevant to point that the impact in runtime of the single or multiple layout floorplans is negligible, while enabling realistic floorplan properties during synthesis and providing a compact starting point for the layout design.

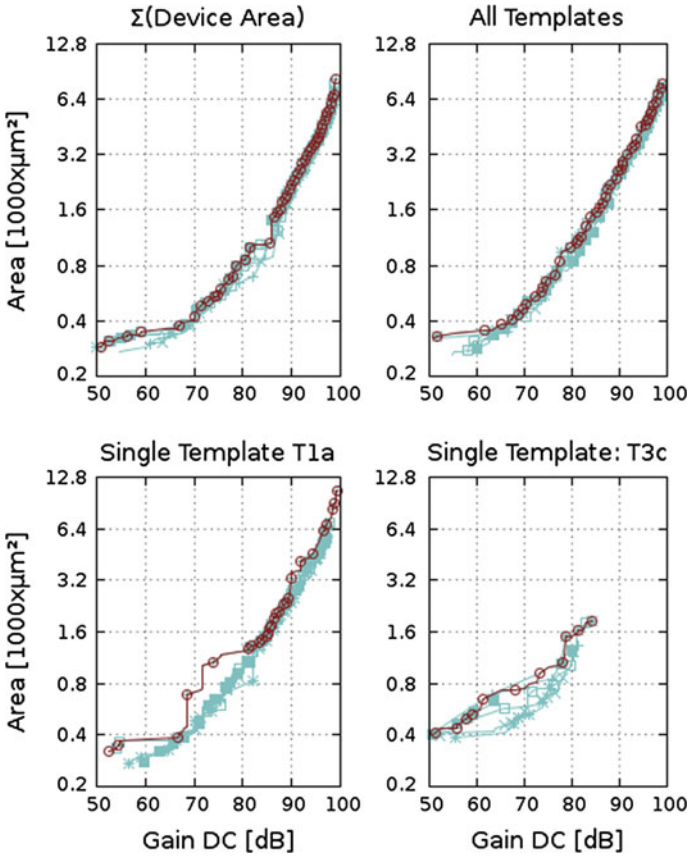


Fig. 7.9 Fronts obtained with 5 runs for the four scenarios highlighting the WCPF. Reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

Also, there are no significant differences in the DC gain range between the results obtained by using multiple floorplans or by considering only the devices area. In the scenario with only T1a there is a slightly larger variation between runs but the ranges of DC gain is still not affected, whereas in the scenario with only T3c, the variability between runs is much wider and there is a clear drop in the achievable DC gain values. By using multiple floorplans the diversity of solutions that is explored increases, thus reducing geometrical over-constraining of the search space that is notorious when just one floorplan is used, leading results that are more reliable.

The WCPFs are plotted together in Fig. 7.10. The most common floorplans in the scenario with the 12 floorplans are T1a, T3a, T2c T2b, and their occurrences are marked in the bottom of the plot with a filled square under the corresponding solution. Not unexpectedly, the solutions obtained considering only the area of the devices dominates the other Pareto fronts, as this unrealistic value assumes an ideal packing without any wasted area. When comparing the WCPFs obtained

Table 7.5 Experimental results for the four scenarios, reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

DC gain aprox.		Min.	56	66	76	86	96	Max.
Sum of devices' area	DC gain (dB)	50.13	56.26	66.56	75.99	85.99	96.04	99.95
	Est. area (μm^2)[a]	285.8	328.1	360.9	636.4	1420.5	4315.6	8251.8
	Area (μm^2)[b]	330.8	382.3	371.6	727.1	1963.3	4887.8	8327.6
	Runtime (min)	50.62						
Single I T1a	DC Gain (dB)	52.66	54.85	66.57	75.63	86.00	96.07	99.27
	Area (μm^2)	323.6	368.7	385.4	1181.1	1781.3	5933.6	10639.1
	Runtime (min)	50.23						
Single II T3c	DC gain (dB)	50.66	55.82	65.16	75.27	84.18	–	–
	Area (μm^2)	401.8	465.1	725.0	995.8	1865.0	–	–
	Runtime (min)	53.24						
All	DC Gain (dB)	50.83	55.14	66.30	76.44	86.36	96.08	99.00
	Area (μm^2)	324.1	341.5	387.3	698.4	1741.1	4957.1	7842.8
	Floorplan	T3a	T1a	T1a	T3a	T2c	T1a	T2c
	Runtime (min)	58.45						

[a]Estimated area is given by the sum of the devices' areas
[b]The area values where obtained by manual placement, post synthesis

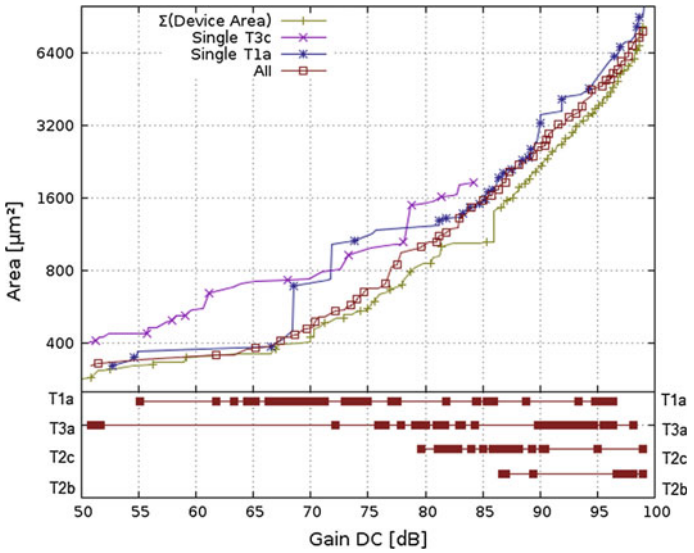


Fig. 7.10 WCPF fronts for the 4 scenarios, identifying the most frequent floorplans in the scenario with the 12 floorplans. Reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

considering realistic floorplans during optimization, the one obtained using multiple floorplans dominates most of the solution space; whereas the WCPF for scenario with just T1a dominates at low DC gain and matches the former in the 83–90 dB gain interval, which is not all that surprising, considering that T1a is the most occurring floorplan in the multi-floorplan front. The solutions obtained with T3c are mostly dominated by all the others.

One of the reasons analog ICs and analog IC design is hard to reuse is their dependence from the surrounding circuitry. To explore this issue and verify the robustness of the proposed approach to specification changes, the output load is replaced by a 10 kΩ resistor in parallel with the same 1 pF capacitor, the biasing current changed to 50 μA and the current consumption limit set to 0.2 mA. The Pareto fronts obtained for 3 executions for same scenarios as before are presented in Fig. 7.11. Again, there are no significant differences in the achievable DC gain

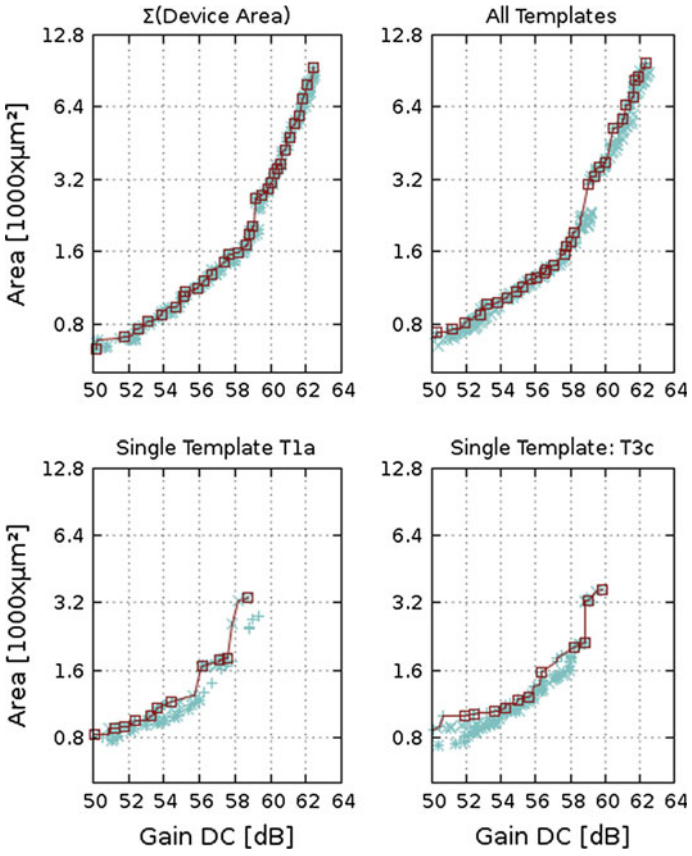


Fig. 7.11 Pareto fronts for the new specifications of the two-stage Miller amplifier, obtained with 3 runs for the four scenarios, highlighting the WCPF. Reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

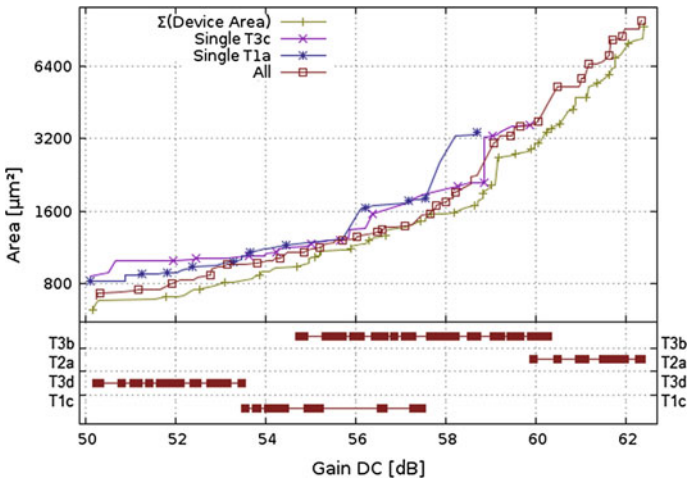


Fig. 7.12 WCPF fronts for the 4 scenarios, identifying the floorplans most used in the scenario with the 12 floorplans for the new specifications. Reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

range between the results obtained using multiple floorplans or considering only the devices areas, i.e., the usage of the floorplan constraints did not limit the search.

In the scenarios with one floorplan there are a few aspects worth noting: first, both optimizations lead to satisfying solutions for smaller DC gain values; second, the range of the DC gain is shorter; and, third, spreading between runs is larger when compared to the scenarios without any floorplan or with multiple floorplans. Particularly, T1a, which was very effective for the previous specifications, is clearly not a good fit for these new specifications.

The WCPFs obtained for the new specifications are plotted together in Fig. 7.12, the most frequent floorplans in the WCPF considering multiple floorplans are T3b, T2a, T2d and T1c with 48, 23, 19 and 17 appearances, respectively. Is relevant to point that although not all combinations appear, all floorplan and partitions are represented. In addition, in this second case the floorplans suitability regions are much better defined.

The floorplans obtained in the different scenarios for DC gain of approximately 55 dB are presented in Fig. 7.13. They show not only different topological arrangements, but also different sizes for the devices that were found during the optimization. The different floorplans impose numerous topological constraints that during the optimization may lead to different solutions having the same area with different DC gain, or for same DC gain, solutions with different areas.

The best layout in terms of area is shown in Fig. 7.13d with an area of 1163.7 μm^2, was obtained in the scenario considering all floorplans. In this layout, two pairs of transistors are drawn in the interdigitized layout style, as specified in the corresponding the layout guides (T3a), and for all the other transistors the folded transistor structure is used with multiple rows to get the form factor that makes the

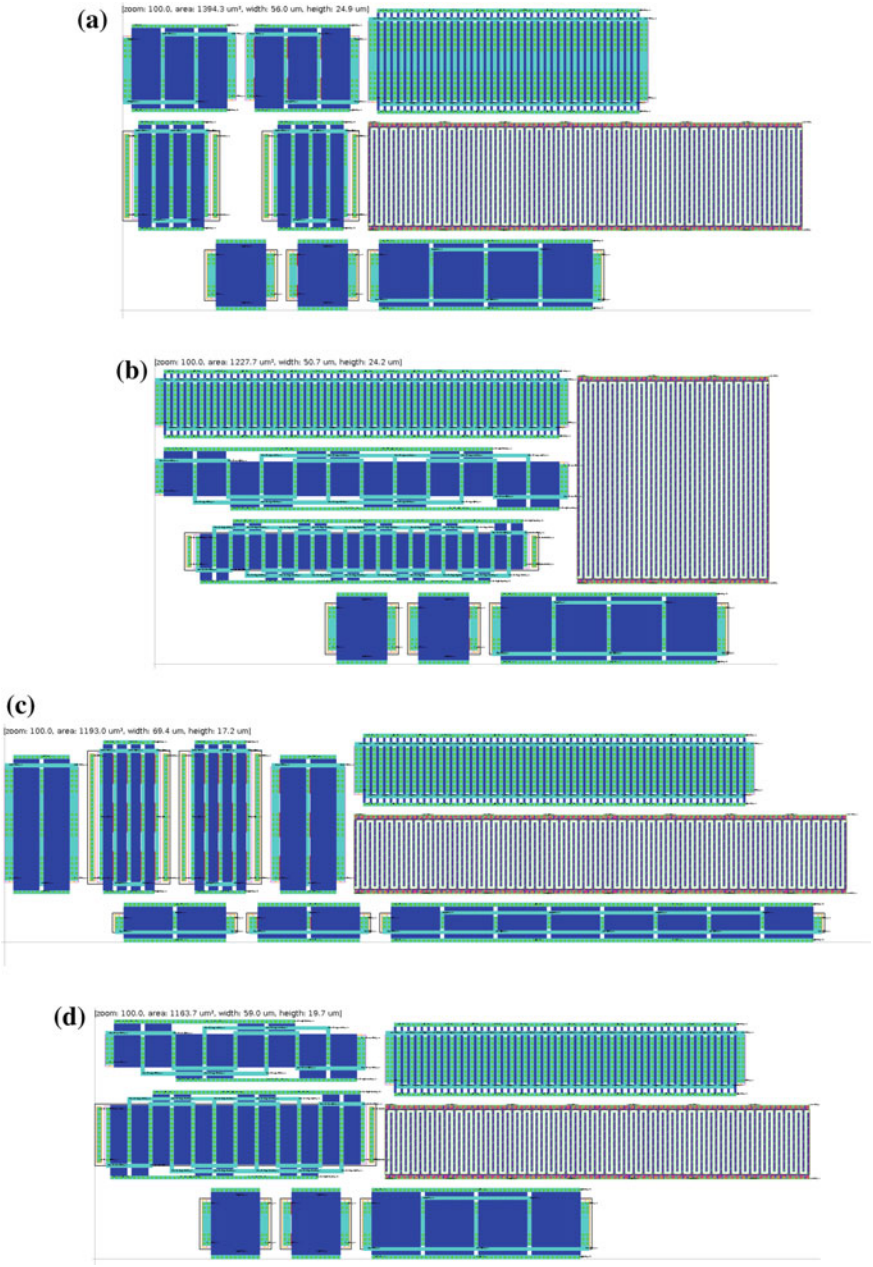


Fig. 7.13 Placement for the designs showing a gain of 55 dB in all scenarios: **a** Sum of devices' area; **b** T1a; **c** T3c; **d** All (all layouts are shown in the same scale). Reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

Table 7.6 Pre and post-layout simulation of the two-stage amplifier designs around 55 dB obtained using the floorplan-aware sizing flow, reprinted from Lourenço [3], Copyright (2015), with permission from Elsevier

Specifications		All[a]		T3c[b]		T1a[c]	
		Pre	Post	Pre	Post	Pre	Post
Gbw	$\geq$ 1 MHz	46.09	41.18	46.29	42.34	36.63	34.48
Gdc	$\geq$ 50 dB	55.04	55.01	55.29	55.24	55.71	55.68
Sr	$\geq$ 0.8 V/µs	130	131	131	131	129	129
Idd	$\leq$ 200 µA	198.9	198.7	199.9	199.5	196.7	196.6
PSRR	$\geq$ 55 dB	60.96	60.70	64.19	65.02	62.71	62.63
Voff	$\leq$ 1 mV	0.57	0.59	0.47	0.45	0.36	0.37
No	$\leq$ 400 μV_{rms}	176.1	167.7	154.8	149.2	149.9	147.1
Sn	$\leq$ 100 nV/$\sqrt{Hz}$	18.06	18.12	15.47	15.52	17.09	17.25
Pm	$\geq$ 55º	55.24	56.45	55.19	55.70	57.53	57.54
ov[4]	$\geq$ 50 mV	79.81	79.71	50.08	50.08	64.11	64.02
ov[5]	$\geq$ 100 mV	100.86	100.89	104.24	104.31	103.25	103.22
D[4, 5]	$\geq$ 100 mV	257.24	257.05	257.24	256.99	246.74	246.60

[a]Post layout measures obtained after routing and extracting the floorplan shown Fig. 7.13d
[b]Post layout measures obtained after routing and extracting the floorplan shown Fig. 7.13c
[c]Post layout measures obtained after routing and extracting the floorplan shown Fig. 7.13b
[d]The constraint applies to: MP11 and MP12, the value shown is the one closest to the specification limit; The constraint applies to MP14, MP20, MN21 and MP22, the value shown is the one closest to the specification limit

best use of the available chip surface. The solutions obtained when not considering floorplan during sizing, considering floorplan T1a and considering T3c have the area of 1394.3, 1227.7, and 1193.7 respectively. As it is easily seen, the differences are not substantial for this range of DC values, and this is especially true for the solutions with smaller devices. But for the larger layouts the impact is relevant, especially if taking into account the dispersion of solutions between the independent runs when only one floorplan is considered. Table 7.6 shows pre- and post-layout simulations results for the solutions obtained using floorplan-aware sizing whose floorplans are shown in Fig. 7.13b–d.

The routing was done using AIDA-L electromigration-aware router [5] and post-layout netlist was extracted with all parasitic capacitors and resistors using Mentor Graphics Calibre tool [6]. The results show that by considering also the routing and its parasitics, small performance variations are obtained, however, all solutions meet specification after the complete layout, showing the validity of the approach and its value for less parasitic dependent designs.

7.2.2 Layout-Aware Sizing

As seen, floorplan-aware sizing uses accurate geometric layout properties during the optimization, and careful layout considerations do help mitigate post-layout effects in the circuit's performance, however, layout parasitic effects are handled implicitly. In this section the results obtained using the layout-aware flow, where the layout parasitic devices are estimated using AIDA-Ls global router, are shown. A revised version of the two stage Miller amplifier is used, considering one additional resistor between the two amplifier stages, is resized for more aggressive bandwidth specifications and considering corners.

In the following tests, the amplifier, loaded with a 10 MΩ resistor in parallel with a 1 pF capacitor and biased with a current of 10 μA, is optimized for minimum area and maximum DC gain. The circuits' performance figures considered and its target performance are indicated in Table 7.7. The optimization variables are again the width, length, number of fingers and number of rows of the MOS devices, the length and width of the resistor, and the length and number of fingers of the MOM capacitor, their ranges are indicated in Table 7.8.

Simultaneously, a floorplan-aware optimization was also performed without considering parasitic-related data. Both the traditional simulation-based and layout-aware runs were started from the same, already sized, solution set reused from the previous optimization (GBW > 50 MHz and IDD < 35 μA), and all solutions were unfeasible for the new target specifications. The first synthesis took around 16 min while the second took around 40 min, and the two resulting POFs of sizing solutions are illustrated in Fig. 7.14.

Table 7.7 Specifications and objectives for the 200 MHz two stage amplifier

Specifications	Measure	Target	Description
Objectives	Gdc (dB)	Maximize	Current consumption
	Area (μm^2)	Minimize	Unity gain frequency
Constraints	Idd (μA)	≤ 200	Current consumption
	Gdc (dB)	≥ 50	Low-frequency gain
	Gbw (MHz)	≥ 200	Unity gain frequency
	Pm (º)	≥ 55	Phase margin
	PSRR (dB)	≥ 55	Power supply rejection ratio
	Voff (mV)	≤ 5	Structural offset voltage
	No (μVrms)	≤ 600	Noise RMS
	Sn (nV/√Hz)	≤ 100	Noise density
	ov[a] (mV)	≥ 50	Overdrive voltages ($V_{GS} - V_{TH}$)[c]
	ov[b] (mV)	≥ 100	Overdrive voltages ($V_{GS} - V_{TH}$)[c]
	d[a, b] (mV)	≥ 100	Saturation margin ($V_{DS} - VD_{Sat}$)[c]

[a]The constraint applies to: MP11 and MP12
[b]The constraint applies to MP14, MP20, MN21 and MP22
[c]For PMOS devices the overdrive is $V_t - V_{gs}$ and delta is $V_{dsat} - V_{ds}$

Table 7.8 Design Variables and Ranges for the for the 200 MHz two stage amplifier

Variable	Min	Max	Grid
lc[a] (µm)	4.4	100	0.1
nfc[a]	14	198	2
lb[b, c, d], lp[e], lal[f], l2g[g] (µm)	0.120	10	0.05
wb[b, c, d], wp[e], wal[f], w2g[g] (µm)	1	10	0.1
nfbp[c], nfb2[d], nfp[e], nfal[f], nf2g[g]	1	200	2
nrb[b, c, d], nrp[e], nral[f], nr2g[g]	1	8	1
lr[h] (µm)	3	60	1
wr[h] (µm)	0.5	1	0.1

[a]The variables lc and nfc are the length and number of fingers of the MOM capacitor XC1
[b]lb, wb and nrb are the length, width and number of rows of MP20
[c]lb, wb, nfbp and nrb are the length, width, number of fingers and number of rows of M14
[d]lb, wb, nfb2 and nrb of MP22
[e]lp, wp, nfp and nrp of MP11 and MP12
[f]lal, wal, nfal and nral of MN9 and MN10
[g]l2, w2,nf2 and nr2 of MN21
[h]The variables lr and wr are the length and width of the Poly resistor MP

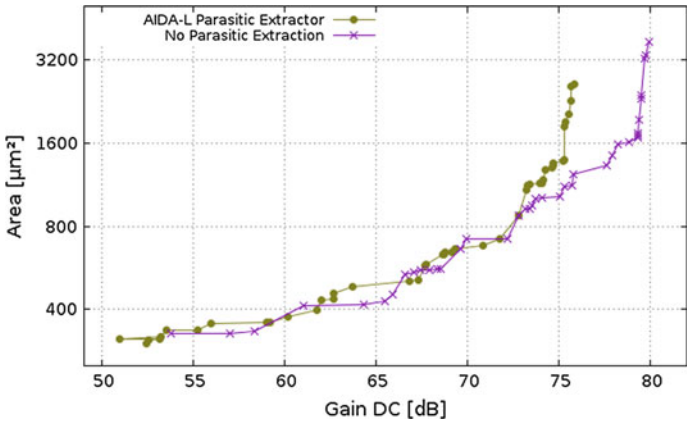


Fig. 7.14 Floorplan and layout-aware optimization POFs

Two design points were selected from the layout-aware POF: the solution with the smallest area showing a GDC around 51 dB, and, a solution showing a GDC of around 75 dB. From the POF without parasitic considerations two similar designs solutions were selected: the one with the GDC closer to 51 dB (with 53.7 dB), and, another with a GDC closer to 75 dB (with 75.4 dB). The layouts for these two designs were automatically generated with AIDA-L [7]. The layouts for the 51 dB designs, the one from the parasitic un-aware flow, the simplified (showing only

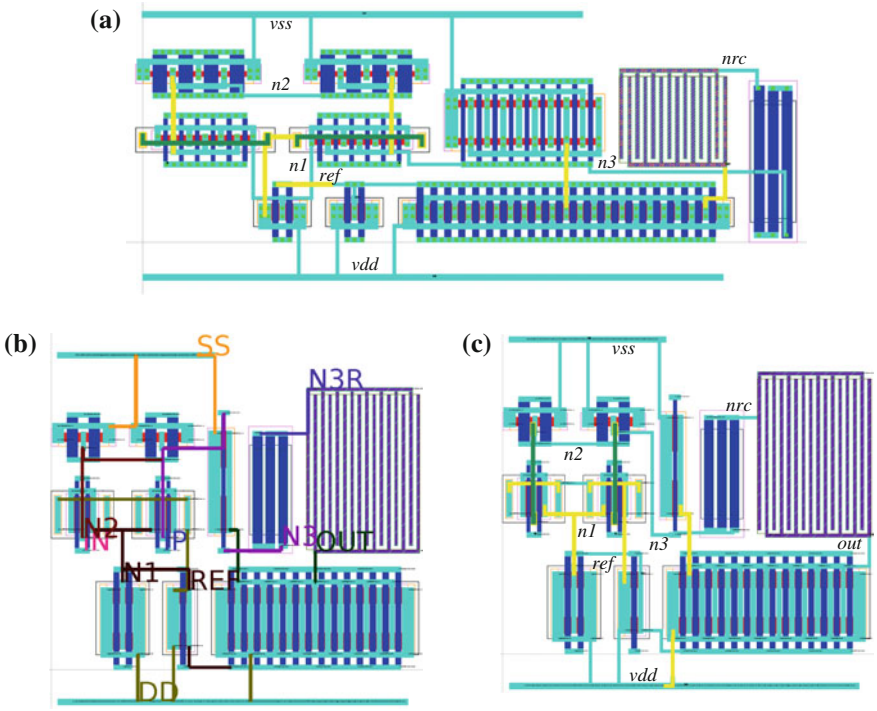


Fig. 7.15 Layout for the 51 dB designs: **a** Layout for the 53 dB solution of the traditional simulation-based sizing, obtained using AIDA-L; **b** Layout obtained using the layout-aware flow, with only global routing for the 51 dB solution. **c** Layout after the detailed routing

global routing) and corresponding detailed layout (after a last step of detailed routing) for the layout-aware flow are shown in (a), (b) and (c) of Fig. 7.15, respectively. Figure 7.16 shows the same for the 75 dBs designs.

The performance measures before and after layout are summarized in Table 7.9. Like in previous efforts on the area, unit-gain frequency and phase margin are the most sensitive measures for typical analog amplifiers [8]. The impact of parasitics is not considered in the traditional or floorplan-aware flow, leading to post-layout designs that do not meet specifications. In the first case (51 dB designs) the area of the resultant layout-aware is smaller than the one obtained by the traditional simulation-based sizing (310 μm^2 against 331 μm^2). While in the second case (75 dB designs), the layout-aware one has a greater area (1330 μm^2 against 1021 μm^2) and is less compact than the layout obtained from traditional optimization-based sizing, however, this was obtained not weighting the impact of the parasitic devices in the solutions' performance. By considering parasitic effects during the entire optimization, what may look overdesign is in fact a feasible solution when considering the parasitic effects. Also, the performance comparisons with Mentor Graphics' Calibre® extractions, a reference verification tool widely

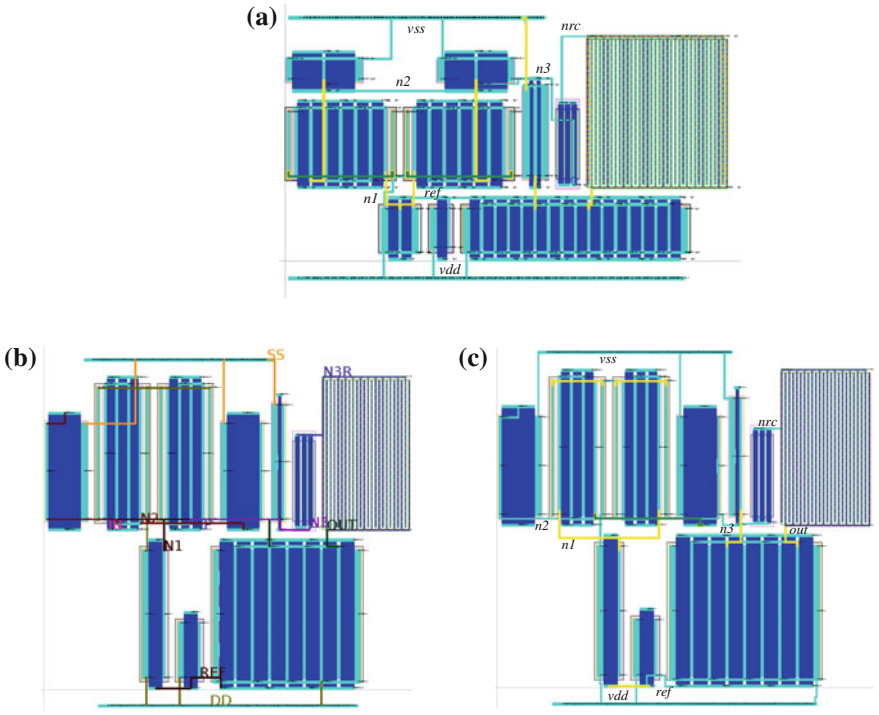


Fig. 7.16 Layout for the 75 dB (**d–f**) designs: **a** Layout for the 75 dB solution of the traditional simulation-based sizing, obtained using AIDA-L; **b** Layout obtained using the layout-aware flow, with only global routing for the 75 dB solution; **c** Layout after the detailed routing

used in industry, confirm the precision of the proposed parasitic extraction over early-stage layout, and its suitability to be included in the layout-aware circuit optimization loop, avoiding the need for in-loop detailed routing.

In the next experiment, the same specifications are considered but additionally to the nominal simulation, the 4 user defined corners, ({FF, −55, 1.26, 0.5}, {SS, 125, 1.14, 0.5}, {FF, −55, 1.26, 0.5}, {SS, 125, 1.14, 0.5} $^{\{\text{Process, Temp., Vdd, VCM}\}}$), are also considered in the evaluation of the circuits' performance. The resulting fronts are shown in Fig. 7.17 and illustrate the behavior of the worst case optimizations, where the solutions whose performance is detailed in Table 7.10 are highlighted.

In this worst case approach, three optimization strategies were followed, first worst case optimization without layout which took 8 h, then an optimization considering both layout and the corner cases which took 32 h, and finally layout effects are included after an initial worst case optimization (4 + 8 h).

In terms of layout effects, again the usage of the proposed layout-aware method leads, in both, cases to solutions whose detailed layout results in feasible solutions, whereas when the layout effects are not considered the post-layout performance does not meets the specification. While the layout generation and parasitic

Table 7.9 Pre/Post-Layout Simulation for nominal case for the 200 MHz two stage amplifier

Specifications		51 dB designs				75 dB designs			
		Traditional		Layout-aware		Traditional		Layout-aware	
		Netlist	Calibre®[b]	PEx[c]	Calibre®[d]	Netlist	Calibre®[e]	PEx[f]	Calibre®[g]
Gdc	max $\geq$ 50 dB	53.7		50.9		75.4		74.6	
Area	min μm^2	337		310		1021		1330	
Gbw	$\geq$200 MHz	208.69	**183.37**[a]	204.29	206.30	206.68	**182.66**[a]	209.25	210.05
Pm	$\geq 55^o$	55.56	**46.42**[a]	74.27	73.54	57.83	58.27	64.51	60.56
Idd	$\leq$200 µA	135.4	135.1	176.8	176.5	190.5	189.2	190.9	190.2
PSRR	$\geq$55 dB	77.23	75.53	75.44	75.45	88.09	77.02	83.89	80.12
Voff	$\leq$5 mV	3.80	3.83	4.39	4.39	4.83	4.91	4.98	4.38
No	$\leq$600 µVrms	435.6	441.8	301.2	305.9	347.7	336.3	381.9	390.3
Sn	$\leq$100 nV/$\sqrt{Hz}$	21.41	21.59	16.17	16.32	25.2	25.7	26.4	26.4

[a]Constraints violated in post-layout performance
[b]Post layout measures obtained after parasitic extraction over the layout in Fig. 7.15a
[c]Post layout measures obtained after parasitic extraction over the floorplan with global routing in Fig. 7.15b
[d]Post layout measures obtained after parasitic extraction over the layout in Fig. 7.15c
[e]Post layout measures obtained after parasitic extraction over the layout shown in Fig. 7.16a
[f]Post layout measures obtained after parasitic extraction over the floorplan with global routing in Fig. 7.16b
[g]Post layout measures obtained after parasitic extraction over the layout shown in Fig. 7.16c

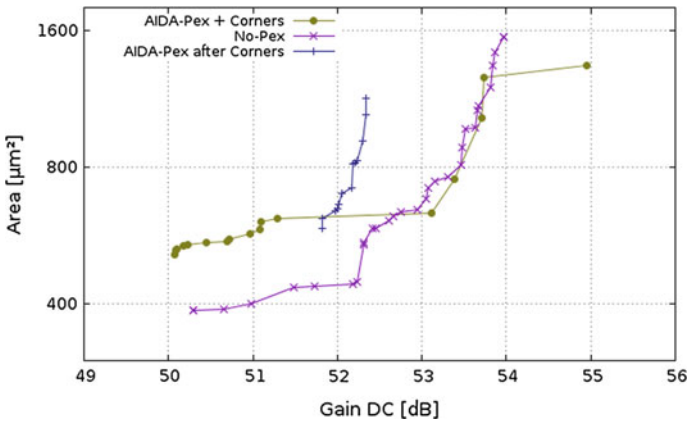


Fig. 7.17 Floorplan, layout-aware and layout-aware after floorplan corner optimization POFs

Table 7.10 Pre/post-layout simulation for worst-case for the 200 MHz two stage amplifier

Specifications		No-PEx		Layout-aware		Layout-aware after corners	
		Netlist	Calibre®	PEx	Calibre®	PEx	Calibre®
Gdc	max $\geq$ 50 dB	50.3		50.1		74.6	
Area	min μm^2	337		512		588	
Gbw	$\geq$ 200 MHz	219.6	201.95	206.57	202.9	200.1	201.89
Pm	$\geq 55^{o}$	55.35	46.86[a]	55.78	56.09	55.99	55.71
Idd	$\leq$ 200 µA	193.5	192.9	182.6	181.3	199.3	198.4
PSRR	$\geq$ 55 dB	60.73	60.66	61.34	61.21	60.16	60.12
Voff	$\leq$ 5 mV	0.61	0.60	0.64	0.64	0.46	0.47
No	$\leq$ 600 µVrms	455.1	457.9	412.5	416.9	421.5	429.9
Sn	$\leq$ 100 nV/$\sqrt{Hz}$	22.08	22.19	22.86	22.99	21.16	21.39

[a]Constraints violated in post-layout performance, all layouts considered were generated using AIDA-L

estimation do increase the execution time, the difference in execution time of the layout-aware approach when compared to the one without layout is also aggravated by the increase in simulation time due to the inclusion of parasitic nodes in the circuit (which was also happening before), but now is multiplied by the number of corners.

In terms of the performance of the circuit, while there is not a large difference between the worst case executions (the layout-aware after corner optimization results in a slightly shorter POF but the DC gain ranges are still similar), however when comparing to the nominal optimization shown before, it is easy to identify the huge reduction in the number of feasible solutions and a much smaller maximum

DC gain. By using multiple floorplans the tools is able to adjust the devices to meet the specifications while compensating for the layout induces parasitic devices without compromising the achievable solutions.

Starting the worst case optimization from the already optimized nominal Pareto set, it is easier to fulfill the additional constraints imposed by the corners, leading to results faster. The same is true for the layout-aware optimization, which was kick started by the output of the worst case optimization. While this approach can strand the optimizer in local minima, it greatly reduces the time required by the tool to get good and complete design, which is also an important aspect for real world applications.

7.3 Two-Stage Folded Cascode Amplifier

The circuit considered in this test case is the two-stage folded cascode amplifier, whose schematic is shown in Fig. 7.18, loaded with a 10 kΩ resistor in parallel with a 1 pF capacitor. The circuit is optimized for maximum bandwidth and minimum current consumption. Once more, both the traditional simulation-based sizing and layout-aware synthesis are done. Both with a population size of 64 for 500 generations, and both started from the same, already sized, solution set. This startup solution set was reused from a similar but different optimization (GBW > 270 MHz and IDD < 500 µA), without layout considerations, and all solutions were unfeasible for the target specifications. The three floorplan templates shown in Fig. 7.19 are considered in the layout-aware optimization.

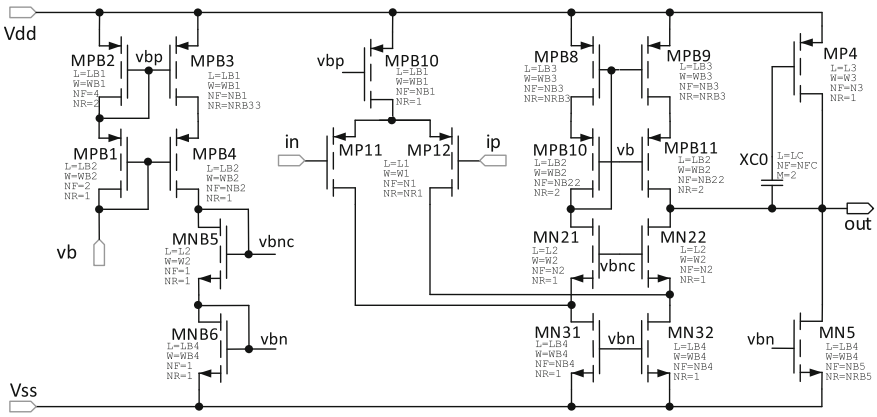


Fig. 7.18 Schematic of the two-stage folded cascode amplifier. Reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

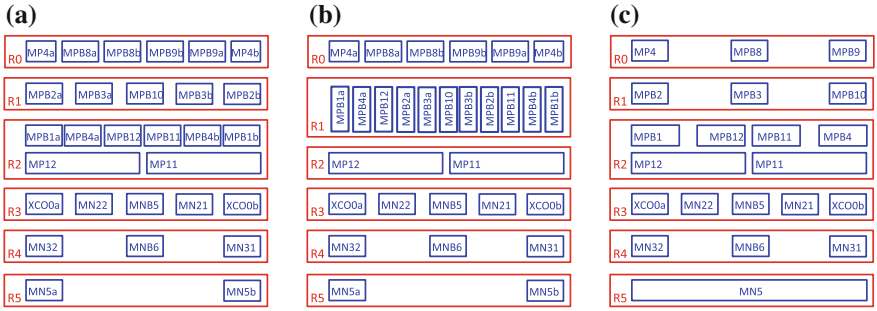


Fig. 7.19 Floorplan templates for the two-stage folded cascode amplifier. Reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

Table 7.11 Variables and ranges for the two-stage folded cascode amplifier

Variable (Unit)	Min.	Grid resolution	Max.
l1, l4, l5, l7, lb7, l9, lb9, l11 (nm)	120	10	10,000
w1, w2, w3, wb1, wb2, wb3, wb4 (μm)	0.24	0.1	30.0
nf1, nf2, nf3, nfb1, nfb2, nfb22, nfb3, nfb4, nfb5	1	2	200
nr1, nrb3, nrb33, nrb5	1	1	20
lc (μm)	4.4	0.1	100
Nfc	14	2	198
Ib (μA)	10.0	10	1000

The optimization variables, listed in Table 7.11, are again the width, length, number of fingers and number of rows of the MOS devices, and the length and number of fingers of the MOM capacitor and the bias current and the target specifications are shown Table 7.12.

In Fig. 7.20, the complete POFs for both traditional and layout-aware synthesis are presented, showing the post layout performance for all solutions. From the typical results, only one held all the specifications (represented by a circle O) in post-layout simulation, all the other became unfeasible (represented by a cross X). The layout-aware POF, although obtaining layouts with a worse performance, guarantees the entire specification fulfillment for all the sizing solutions.

The first synthesis took around 1 h while the second took around 1.5 h. In Fig. 7.21a, b the resulted layouts for the most power efficient circuits found by both methods are illustrated. Although the layouts are alike, the traditional one does not fulfill the specifications as the layout-aware sizing approach does. Both circuits' performances, simulated using the netlist extracted both from AIDA parasitic extractor and from CALIBRE®, are presented in Table 7.12.

Table 7.12 Performance comparison for the traditional and layout-aware optimizations, reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

Specifications		Traditional			Layout-aware	
		Netlist[a]	PEx	Calibre®	PEx	Calibre®
Idd	Min, (mA)	5.35			5.62	
GBW	Max, $\geq$400 MHz	402.07	392.33	392.59	408.05	408.2
GDC	$\geq$70 dB	73.53	73.53	73.53	75.68	75.68
Area	$\leq$20 k μm^2	12.09 k	16.49 k		16.34 k	
Noise	$\leq$500 µVrms	197.05	195.37	194.93	195.48	195.16
PM	$\geq$55º	57.97	55.54	55.26	57.04	56.78
Ov[b]	$\geq$50 mV	59.48	59.48	59.48	50.61	50.48
D[c]	$\geq$100 mV	100.33	105.06	105.06	105.37	105.06

[a]Performance for the netlist excluding parasitics
[b]Overdrive voltage
[c]Saturation margin

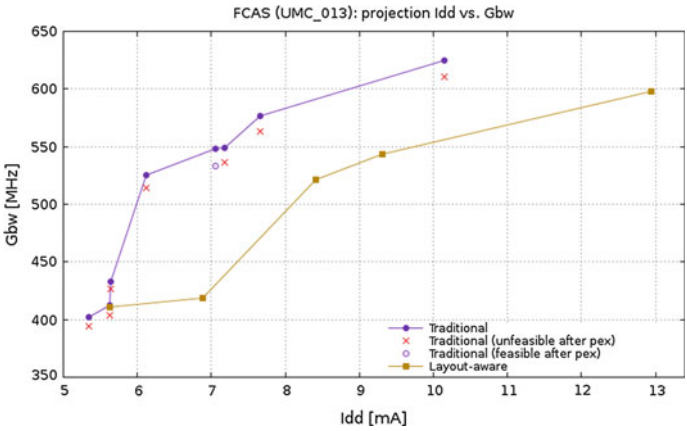


Fig. 7.20 Traditional and Layout-aware optimization POFs. Reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

In the traditional simulation-based optimization the parasitic effects are not accounted for, and for that reason, when the resulted layout is simulated with the inclusion of circuit's parasitics, the GBW specification is no longer met. In the layout-aware sizing optimization, the circuit is wittingly over estimated for nominal simulation, so that when the parasitic components are added the specifications are all fulfilled. In terms of consumption the circuit needs slightly more current, consequently, increasing the frequency to hold to the specification in the presence of layout parasitic effects. Moreover, the comparison between the performance

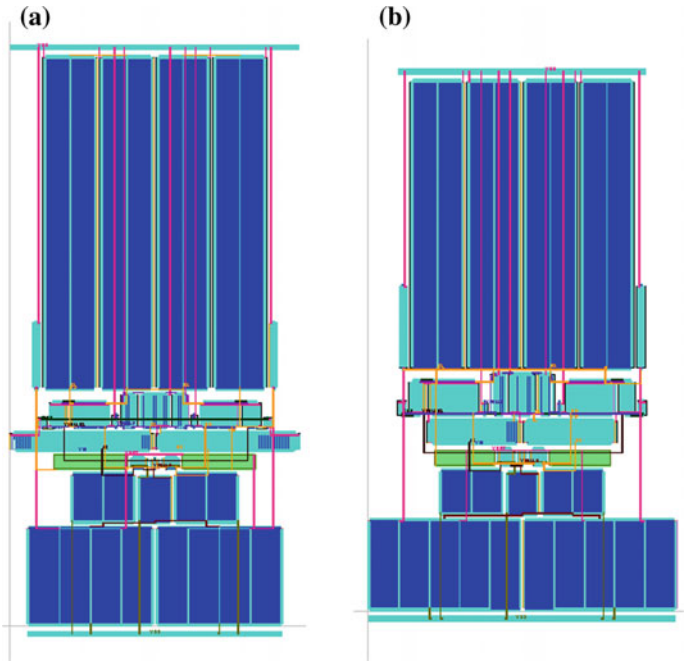


Fig. 7.21 **a** Layout obtained from traditional design (fails specs in post layout) and **b** Layout obtained from layout-aware (post-layout correct). Reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

measurements from the AIDA parasitic netlist and the ones obtained using an industry reference extraction tool, show that in these bandwidths there is a good enough matching.

7.4 Single Stage Amplifier with Gain Enhancement Using VCs

The single stage amplifier with gain enhancement using voltage combiners (VCs) proposed in [9] loaded with a 6 pF capacitor, introduced in Chap. 5 of this book and (re)show here for convenience, is considered in this test case in two ways: (1) a floorplan-aware optimization is done to improve the floorplan of a manually sized design; (2) layout-aware sizing is executed. The circuit schematic is shown in Fig. 7.22. The design variables and ranges are defined in Table 7.13 and the specifications are presented in Table 7.14. In addition, the following four corner conditions are considered for devices' models: slow NMOS and slow PMOS (SS); slow NMOS and fast PMOS (SNFP); fast NMOS and slow PMOS (FNSP); and, fast NMOS and fast PMOS (FF).

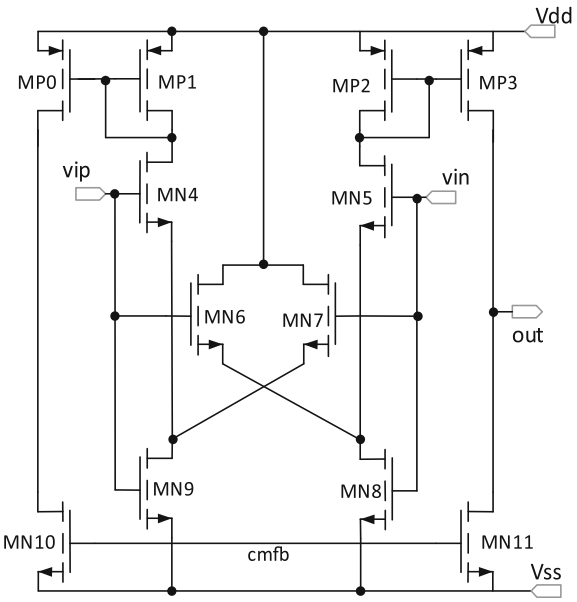


Fig. 7.22 Schematic of the single stage amplifier with gain enhancement using VCs. Reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

Table 7.13 Variables and ranges for the single stage amplifier with gain enhancement using VCs, reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

Variable (unit)	Min.	Grid resolution	Max.
l0, l1, l4, l6, l8, l10 (nm)	120	10	1000
w0 w1 w4 w6 w8 w10 (μm)	1	0.1	10
nf0, nf1, nf4, nf6, nf8, nf10	1	2	8

The variables l0, w0, and nf0 are dimensions of MP0 and MP3; l1, w1, and nf1 of MP1 and MP2; l4, w4 and nf4 of MN4 and MN5; l6, w6 and nf6 of MN6 and MN7; l8, w8, and nf8 of MN8 and MN9; l10 w10 and nf10 of MN10 and MN11

7.4.1 Floorplan-Aware Sizing

This test case compares the manual layout with a layout created by the AIDA-C floorplan-aware optimization. The layout guides used in this floorplan-aware optimization are shown in Fig. 7.23a and the manual layout that will be used for comparison is presented in Fig. 7.24b.

The optimization was done to obtain, approximately, the same figure of merit (FOM), 1150, and the same gain, 50 dB, under the corner conditions of the the original design [9]. The floorplan-aware optimization took approximately 45 min, and was started from the Pareto front obtained in [9], always considering the PVT corners. Figure 7.24 shows a screenshot of AIDA, showing the solution that was found by the tool.

Table 7.14 Specifications for the single stage amplifier with gain enhancement using VCs, reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

Specifications	Measure	Target	Units	Description
Constraints	Idd	≤ 350	µA	Current consumption
	Gdc	≥ 50	dB	Low-frequency gain
	Gbw	≥ 30	MHz	Unity gain frequency
	Pm	≥ 60	°	Phase margin
	FOM	≥ 1000	MHz pF/mA	Figure of merit $FOM = \frac{Gbw \times C_{Load}}{Idd}$
	ov[a]	≥ 50	mV	Overdrive Voltages $(V_{GS} - V_{TH})$[c]
	ov[b]	≥ 100	mV	Overdrive Voltages $(V_{GS} - V_{TH})$[c]
	d[a, b]	≥ 50	mV	Saturation margin of the PMOS device $(V_{DS} - V_{DSat})$[c]

[a]Applies to: MP0, MP1, MP2 and MP3
[b]Applies to: MN4, MN5, MN6, MN7, MN8, MN9, MN10 and MN11
[c]For PMOS devices the overdrive is $V_t - V_{gs}$ and delta is $V_{dsat} - V_{ds}$

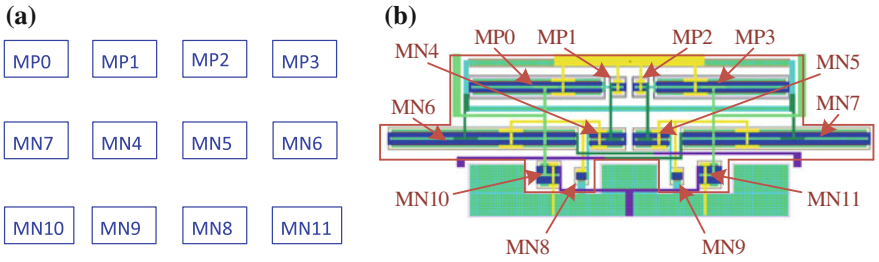


Fig. 7.23 Floorplan template and manual layout for the single stage amplifier with gain enhancement using VCs: **a** Floorplan template; **b** Manual layout

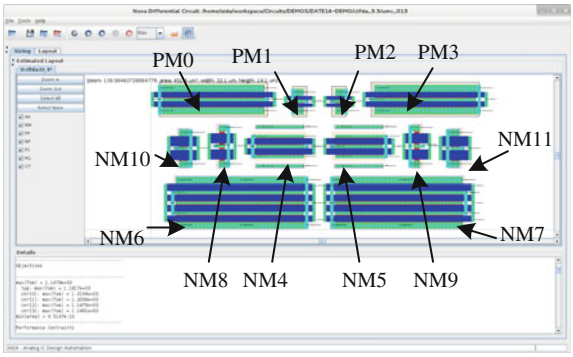


Fig. 7.24 AIDA screenshot showing the solution obtained using the floorplan-aware sizing

AIDA-C floorplan-aware optimization decreased the area of the floorplan from 646 μm^2 in the manual layout (inside the frame), to 451.5 μm^2 in the automatically generated one. In this manner the layout design phase is accelerated with the provided starting point, which already satisfies the guidelines of the designer and whose sizes where already adjusted to increase packing. The layout created by AIDA-C does not have routing implemented but is still a fair comparison, as the bulk/substrate polarizations contacts are included and devices are placed at a distance that is capable of accommodating the metal strips to make the needed connections.

In this example the reduction of 30 % in the area is obtained by the effective exploration of alternative transistor structures during the optimization. The optimizer automatically considered more fingers in the longer transistors by folding the transistors and adjusted widths and lengths to create a compact layout. These "geometrical" device parameters (number of fingers and number of rows) impact strongly in the obtained area than the "design" parameters (e.g. width and length), nevertheless changing the geometrical parameters does modify the device, and as such have impact in the devices' performance and can be easily included in the simulation to increase the simulation results accuracy. Therefore, AIDA-C, also considers the "design" parameters in the optimization to compensate the performance degradation that may occur. The major contribution of AIDA-C to the area decrease is not because a better packing of the original device sizes, but AIDA-C optimized the device's sizes to pack better while keeping the design constraints in check.

7.4.2 Layout-Aware Sizing

In this Section, the circuit is optimized to maximize bandwidth (GBW) and minimize both current consumption (Idd) and area. The circuit's variables and specifications are the same as before, but the set of floorplan templates shown in Fig. 7.25 is considered instead.

A three step optimization was followed considering in each step optimizations with a population size of 128 for 200 generations. First an optimization considering only nominal conditions, without corners or layout parasitic effects, which took 15 min was done. Then, an optimization considering the corner cases, which took 1 h, and was started form the results of the previous optimization. And finally, the layout effects are included after the previous corner optimization, in a worst case and layout-aware (WC&LA) optimization, which took 1 h and 30 min, on a total of 2 h and 45 min for the complete sizing optimization. The resulting Pareto sets are shown in Fig. 7.26 and illustrate the behavior of the three optimization stages, where the solutions whose performance is detailed in Table 7.15, are highlighted in the 3D scatter plot and in the two 2D projections. The corresponding layouts of these three solutions are shown in Fig. 7.27.

The nominal Pareto set was found much faster, and dominates the others, but it is neither robust to the variability of the fabrication process or to layout induced parasitic

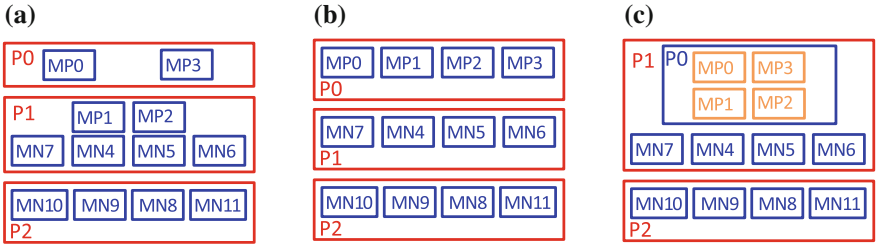


Fig. 7.25 Floorplan templates for the single stage amplifier with gain enhancement using VCs. Reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

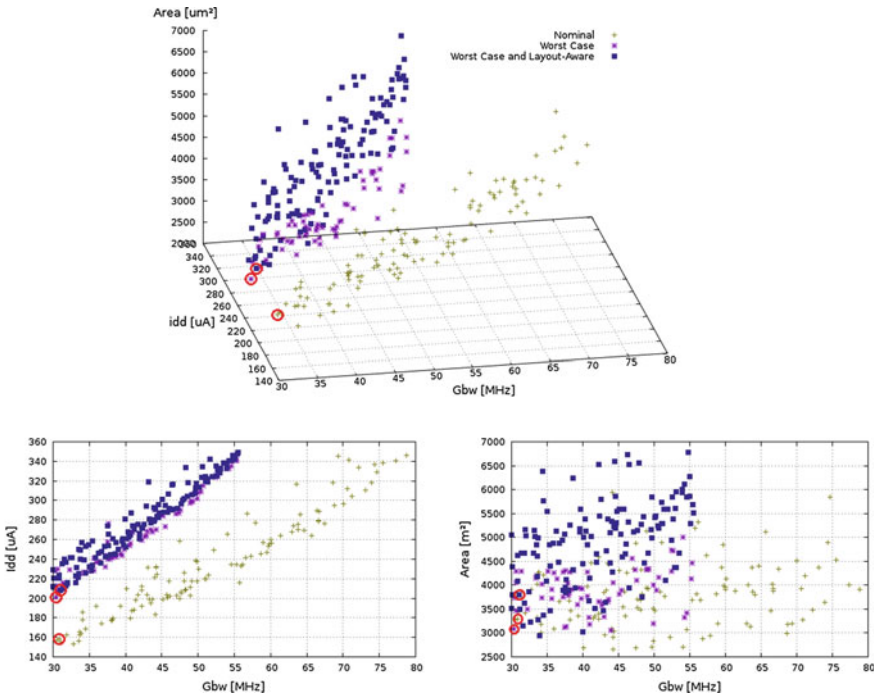


Fig. 7.26 Pareto sets for the single stage amplifier with gain enhancement using VCs. Reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

devices. The two worst case approaches are similar with respect to the electrical performance (Idd and GBW) and, as in the previous example, show meaningful reduction in the circuit's performance, as no solutions with GBW larger than 55 MHz or Idd smaller than 200 μA were found. Likewise, the one ignoring layout's parasitic effects is slightly better with respect to the area. The performance of the solution with smaller power consumption obtained in each stage is shown in Table 7.15. To illustrate the previously said, the performance of each design, evaluated using the more accurate and robust methods, is also included. Showing that the solutions from

Table 7.15 Performance of the single stage amplifier for solution with lowest power in each optimization stage, reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

Specifications		Nominal			Worst-case		WC and LA
		Nominal	WC[a]	WC&LA[a]	WC	WC&LA[a]	
Idd (μA)	min $\leq$ 350	156.8	188.2	188.3	200.8	201.1	208.1
Gbw (MHz)	max $\geq$ 30	30.6	25.3[b]	24.7[b]	30.4	29.7[b]	31.1
Area (μm^2)	Min	3274			3089		3807
Gdc (dB)	$\geq$ 50	54.8	54.2	54.1	51.1	51.1	51.1
Pm (º)	$\geq$ 60º	79.6	77.1	73.8	81.8	78.6	83.5
FOM (MHz pF/mA)	$\geq$ 1150	1168	1143	1107	1234	1199	1295

[a]Circuits performance evaluated using considering stricter conditions
[b]Constraints violated, all layouts considered were generated using AIDA-L

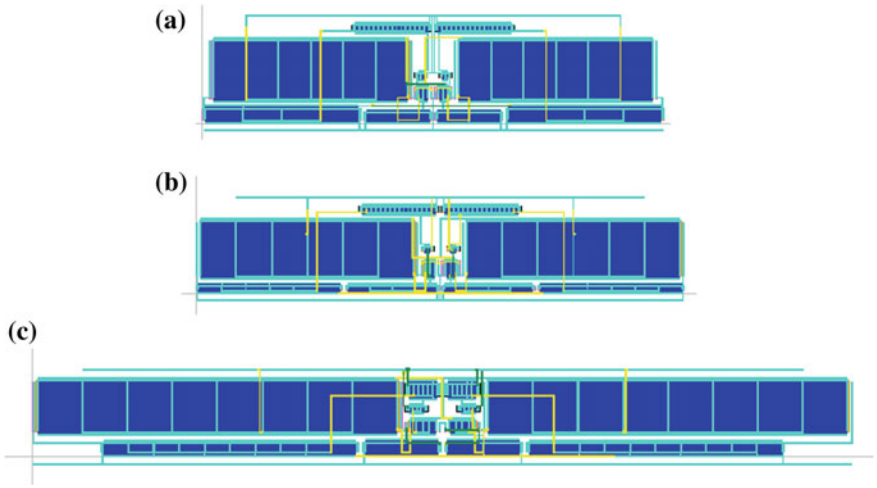


Fig. 7.27 Layout for the single stage amplifier solutions shown in Table 7.15. **a** Nominal, **b** worst case, **c** worst-case and layout-aware. Reprinted from Lourenço [2], Copyright (2016), with permission from Elsevier

the first and second stages fail to meet the specification, the first showing a significant 16.67 % error, while the second shows a small error (2.3 %).

In terms of the solutions found by the optimizer, it was easy to see that while the less constrained problem presents wide solution ranges, the inclusion of worst case corners greatly reduces the spread feasible solutions, cutting chunks of the nominal POF away. This did not happened with the inclusion of the layout effects alone, as the tool adjusts the sizes of the devices to compensate for the layout parasitics without sacrificing the spread of the solutions. The use of multiple floorplans greatly contributes for this effect. In this context, the results show that the layout-aware sizing optimization is of the utmost importance for effective and

reliable automatic layout generation, leading to automatically generated designs that meet all specifications in post-layout simulations. However, for characterization of the performance landscape, e.g., if Pareto fronts are to be used to model circuits for system-level synthesis, variability is the most relevant factor and cannot be overlooked in layout-aware sizing solutions.

7.5 Conclusion

The proposed approach was validated with both classical and new analog circuit structures showing its generality. The inclusion of layout effects increases the effectiveness of the sizing to layout flow. The fact that layout aware synthesis was made, eases the transfer of the AIDA-C output to AIDA-L input, yielding final layouts that are well packed without designer intervention, moreover, since multi-objective optimization was used and the layout estimation is fairly accurate and immediate (milliseconds), the designer intervention can focus on the efficiently exploration of design tradeoffs.

The developed layout-aware synthesis methodology for automatic sizing of analog ICs addresses some drawbacks of the existing state-of-the-art approaches. It merges the circuit optimizer with the layout generator, which offers an innovative solution for parasitic estimation of interconnects by computing the optimal electrical-current correct WT and global routing in-loop for each different sizing solution. The light-weight built-in extractor estimates the impact of layout parasitics for both floorplan and early-stages of routing without requiring a detailed layout, greatly reducing overall evaluation time to in parasitic-aware optimization. In the circuits considered so far the AIDA parasitic estimates closely follow the ones obtained from CALIBRE®. Nonetheless, if the approximated routing would lead to misleading estimated parasitic effects, one additional step using a commercial extractor after the detailed routing can be considered as the automatic tool wraps up the design.

Moreover, as the routing template/setup is not fixed, not only the parasitic-aware performances and geometric requirements are considered, but also, the routing topology follows the optimization process.

References

1. Martins R, Lourenco N, Canelas A, Povoa R, Horta N (2015) AIDA: Robust layout-aware synthesis of analog ICs including sizing and layout generation. In: International conference on synthesis, modeling, analysis and simulation methods and applications to circuit design (SMACD), Istanbul, 7–9 Sept 2015
2. Lourenço N, Martins R, Canelas A, Póvoa R, Horta N (2016) AIDA: Layout-aware analog circuit-level sizing with in-loop layout generation. doi:10.1016/j.vlsi.2016.04.009
3. Lourenço N, Canelas A, Póvoa R, Martins R, Horta N (2015) Floorplan-aware analog IC sizing and optimization based on topological constraints. Integr VLSI J Elsevier 48:183–197

4. Lourenco N, Martins R, Horta N (2015) Layout-aware sizing of analog ICs using floorplan & routing estimates for parasitic extraction. In: 2015 Design, automation & test in Europe Conference & Exhibition (DATE), Grenoble, 9–13 March 2015
5. Martins R, Lourenço N, Canelas A, Horta N (2014) Electromigration-aware analog Router with multilayer multiport terminal structures. Integr VLSI J 47(4):532–547
6. Mentor Graphics (2014) Mentor Graphics. http://www.mentor.com. Accessed 23 Nov 2014
7. Martins R, Lourenço N, Horta N (2013) LAYGEN II—Automatic Layout Generation of Analog Integrated Circuits. IEEE Trans Comput-Aided Des Integr Circuits Syst (TCAD) 32 (11):1641–1654
8. Castro-Lopez R, Guerra O, Roca E, Fernandez FV (2008) An integrated layout-synthesis approach for analog ICs. IEEE Trans Comput Aided Des Integr Circuits Syst 27(7):1179–1189
9. Póvoa R, Lourenço N, Horta N, Santos-Tavares R, Goes J (2013) Single-stage amplifiers with gain enhancement and improved energy-efficiency employing voltage-combiners. In: 21st IFIP/IEEE international conference on very large scale integration, Istambul, 2013

Chapter 8
Conclusions

8.1 Conclusions

The multi-objective nature of the IC design synthesis makes it well suited for automatic design using multi-objective optimization strategies. In the approach taken, the output is not one solution, but a set of completely designed non-dominated solutions, all meeting the specifications. It is up to the designer to use system level criteria to select the tradeoff between the concurrent objectives that better suits the target project. The inclusion of corner cases and the usage of industrial circuit simulators ensure an automatic circuit level sizing that is compliant with the requirements of the analog designer. The usefulness of AIDA-C to designers was shown using different design strategies. First, using the Typical or nominal (T) design strategy, the designer explores several design tradeoffs in a matter of minutes, which is useful for system level design and tradeoff landscaping. Then, using the Corners (C) or TC strategies the designer can obtain a family of optimum robust circuits that comply with the specification in all corner cases considered.

By embedding a simple but effective design knowledge model, the Gradient Model, into the evolutionary optimization kernel of the optimizer efficiency is enhanced, forwarding the data to the desired objectives and causing a significant reduction in the required number of evaluations, i.e., electrical simulations. Moreover, the simplicity of the model makes its generation extremely easy and fast. The advantages of the approach were proved for typical analog circuits by making a statistical analysis of the produced results.

By efficiently linking AIDA-C with AIDA-L, the estimated parasitic devices are fed to AIDA-C that re-sizes the circuit components to compensate layout effects parasitics. The inclusion of layout characteristics in the sizing flow reduce the number of iterations between the electrical and physical design stages and enables a fully automatic sizing to layout flow that closes the traditional analog IC design flow.

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0_8

Finally, and most of the times forgotten, designer interface and helper functionalities are of utmost importance, even in a research environment as the proof of applicability depend not only of the quality of the results but also on how easy it is to use. AIDA-C is being used by the industry for a year now, and the clear interface was extremely relevant to the effectiveness of this computer-aided design (CAD) approach, so that the user (analog designer) use the tool without trading the time saved for the sizing optimization, with the time he takes to set it up. Moreover, an efficient setup flow eases reuse and migration of designs to different specifications and technologies.

8.2 Future Work

Figure 8.1 shows how the proposed methodology is framed in the design flow. However the tool is neither perfect nor complete, in this section; possible future directions for the development and research in AIDA are outlined.

- **Inclusion of Knowledge to Accelerate Sizing**: The gradient model has shown how, even simple models that bring knowledge into the optimization, can increase the quality of the solution or decrease the time to get it. The exploration of new automatic learning techniques to further incorporate automatically extracted knowledge in the optimization can be studied and explored. Surrogate-based evaluation had been use with some success in pruning the

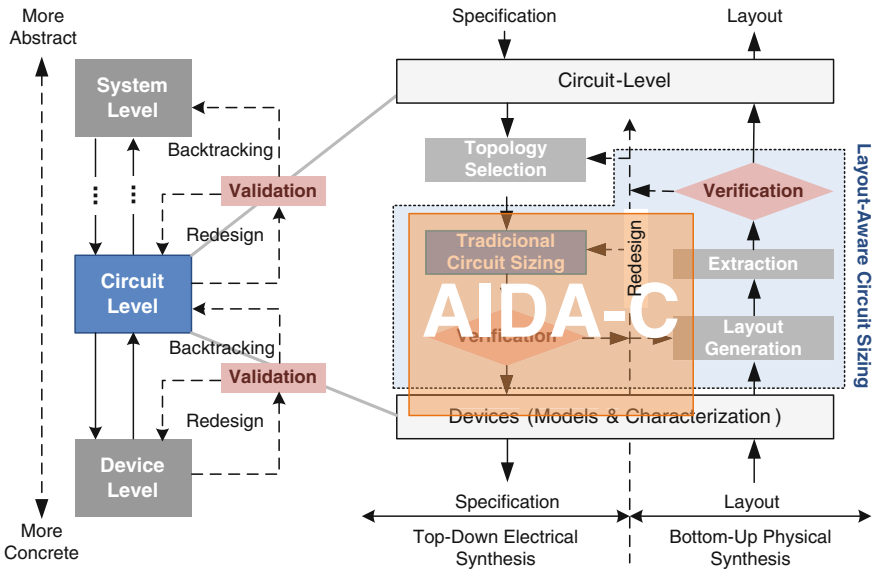


Fig. 8.1 AIDA-C in analog IC design flow

number of expensive evaluation during optimization, the implementation of a surrogate assisted optimization kernel may lead to performance improvements.

- **Variability and Layout Effects**: Variability-aware and yield-optimization sizing, i.e., sizing that takes into account the randomness in the design process, is of the utmost relevance, and like corner analysis, this is a commonly used technique. The massive number of required simulations is a detrimental property. The currents state-of-the-art all approaches focus on improving the yield locally (around one solution), to verify how these approaches can be extended to handle sets of different solutions, like the Pareto fronts, efficiently is also an interesting research subject. Besides, both layout parasitic effects and yield degradation due to process variation (environment variations such as power and temperature variations are also relevant) are considered in the current state of the art separately, but are not unrelated. For example a common technique to reduce the impact of variation is to increase the size of the devices, but at the expense of also increasing those devices parasitic capacitances. Instead of treating layout parasitic effects and process variation separately, both non-idealities could be handled simultaneously.
- **Constraint Handling**: In addition new approaches to adaptive constraint handling for the increasing complexity of analog and mixed systems that bring together the tool and the designer are of the utmost relevance to disrupt the current state of mistrust in automation tools by analog designers. Design specifications are often contradictory which makes them difficult to handle automatically. It is not unusual for a designer to over-constrain the problem in the early stages of development, and then, facing the impossibility of finding a solution, re-evaluate the specifications, with changes in: topology, such that the new topology meet the specifications; new system level architectures that can handle the circuit's reduced performance; or even simply to accept the block's failed specification (if the specification is non-critical and the obtained performance figure is not far for what was required). Therefore, besides the accurate circuit performance evaluation including layout parasitic and variability considerations, for the analog designer to be comfortable using these automations tools, an efficient adaptive constraints handling scheme must be researched and developed.
- **Topology Selection/Generation**: Having the right topology for a given set of specifications is indispensable for a high performance design. Several topology selection and generation schemes were proposed by the scientific community. However, designers are suspicious of generated structures; hence the topology selection is still mostly based on designers experience and trust that a given topology will perform well for the given set of specifications. Multi-objective optimization techniques ease the assessment of optimum design tradeoffs that meet the specifications, by extending the implemented solution to include multiples topologies; it would enable a more comprehensive tradeoff analysis.

Index

N. Lourenço et al., *Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects*, DOI 10.1007/978-3-319-42037-0