Computer-Aided Design of Analog and Mixed-Signal Integrated Circuits

GEORGES G. E. GIELEN, SENIOR MEMBER, IEEE, AND ROB A. RUTENBAR, FELLOW, IEEE

Invited Paper

This survey presents an overview of recent advances in the state of the art for computer-aided design (CAD) tools for analog and mixed-signal integrated circuits (ICs). Analog blocks typically constitute only a small fraction of the components on mixed-signal ICs and emerging systems-on-a-chip (SoC) designs. But due to the increasing levels of integration available in silicon technology and the growing requirement for digital systems to communicate with the continuous-valued external world, there is a growing need for CAD tools that increase the design productivity and improve the quality of analog integrated circuits. This paper describes the motivation and evolution of these tools and outlines progress on the various design problems involved: simulation and modeling, symbolic analysis, synthesis and optimization, layout generation, yield analysis and design centering, and test. This paper summarizes the problems for which viable solutions are emerging and those which are still unsolved.

Keywords*—Analog and mixed-signal computer-aided design (CAD), analog and mixed-signal integrated circuits, analog circuit and layout synthesis, analog design automation, circuit simulation and modeling.*

I. INTRODUCTION

The microelectronics market and, in particular, the markets for application-specific ICs (ASICs), application-specific standard parts (ASSPs), and high-volume commodity ICs are characterized by an ever-increasing level of integration complexity, now featuring multimillion transistor ICs. In recent years, complete systems that previously occupied one or more boards have been integrated on a few chips or even one single chip. Examples of such **systems on a chip** (**SoC**) are the single-chip TV or the single-chip camera [1] or new generations of integrated telecommunication systems that include analog, digital, and eventually radio-frequency (RF) sections on one chip. The technology of choice for these systems is of course CMOS, because of the good digital scaling, but also BiCMOS is used when needed for the analog or RF circuits. Although most functions in such integrated systems are implemented with digital or digital signal processing (DSP) circuitry, the **analog circuits needed at the interface between the electronic system and the "real" world** are also being integrated on the same die for reasons of cost and performance. A typical future SoC might look like Fig. 1, containing several embedded processors, several chunks of embedded memory, some reconfigurable logic, and a few analog interface circuits to communicate with the continuous-valued external world.

Despite the trend previously to replace analog circuit functions with digital computations (e.g., digital signal processing in place of analog filtering), there are **some typical functions that will *always* remain analog**.

- The first typically analog function is on the input side of a system: signals from a sensor, microphone, antenna, wireline, and the like, must be sensed or received and then amplified and filtered up to a level that allows digitization with sufficient signal-to-noise-and-distortion ratio. Typical analog circuits used here are low-noise amplifiers, variable-gain amplifiers, filters, oscillators, and mixers (in case of downconversion). Applications are, for instance, instrumentation (e.g., data and biomedical), sensor interfaces (e.g., airbag accelerometers), process control loops, telecommunication receivers (e.g., telephone or cable modems, wireless phones, set-top boxes, etc.), recording (e.g., speech recognition, cameras), and smart cards.
- The second typically analog function is on the output side of a system: the signal is reconverted from digital to analog form and it has to be strengthened so that it can drive the outside load (e.g., actuator, antenna, loudspeaker, wireline) without too much distortion. Typical analog circuits used here are drivers and buffers, filters, oscillators and mixers (in case of upconversion). Applications are, for instance, process control

Manuscript received February 4, 2000; revised July 14, 2000.

G. G. E. Gielen is with the Electrical Engineering Department, Katholieke Universiteit Leuven, Leuven, Belgium.

R. A. Rutenbar is with the Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh, PA 15213 USA.

Publisher Item Identifier S 0018-9219(00)10757-1.

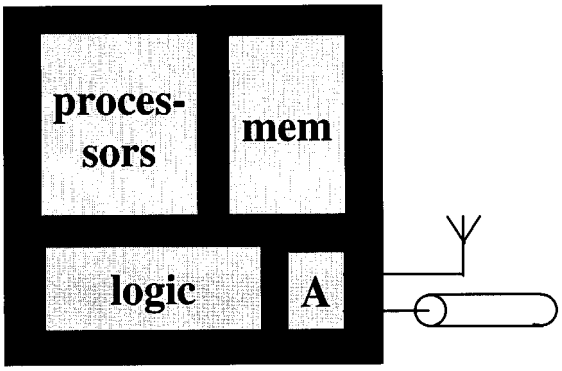

Fig. 1. Future system-on-a-chip.

loops (e.g., voltage regulators for engines), telecommunication transmitters, audio and video (e.g., CD, DVD, loudspeakers, TV, PC monitors, etc.), and biomedical actuation (e.g., hearing aids).

- The third type of blocks are the true mixed-signal circuits that interface the above analog circuits with the DSP part of the system. Typical circuits used here are the sample-and-hold circuits for signal sampling, analog-to-digital converters for amplitude discretization, digital-to-analog converters for signal reconstruction, and phase-locked loops and frequency synthesizers to generate a timing reference or perform timing synchronization.
- In addition, the above circuits need stable absolute references for their operation, which are generated by voltage and current reference circuits, crystal oscillators, etc.
- Finally, the largest analog circuits today are high-performance (high-speed, low-power) digital circuits. Typical examples are state-of-the-art microprocessors, which are largely custom sized like analog circuits, to push speed or power limits.

Clearly, analog circuits are indispensable in all electronic applications that interface with the outside world and will even be more prevalent in our lives if we move toward the intelligent homes, the mobile road/air offices, and the wireless workplaces of the future.

When both analog (possibly RF) and digital circuits are needed in a system, it becomes obvious to integrate them together to reduce cost and improve performance, provided the technology allows us to do so. The growing market share of integrated mixed-signal ICs observed today in modern electronic systems for telecommunications, consumer, computing, and automotive applications, among many others, is a direct result of the need for higher levels of integration [2]. Since the early 1990s, the average growth rate of the mixed-signal IC market has been between 15% and 20% per year, and this market is predicted to surpass $22 billion by 2001. Recent developments in CMOS technology have offered the possibility to combine good and scalable digital performance with adequate analog performance on the same die. The shrinking of CMOS device sizes down to the deep submicrometer regime (essentially in line with, or even ahead of, the predicted technology roadmap [3]) makes higher levels of system integration possible and also offers analog MOS transistor performance that approaches the performance of a bipolar transistor. This explains why CMOS is the technology of choice today, and why other technologies like BiCMOS are only used when more aggressive bipolar device characteristics (e.g., power, noise, or distortion) are really needed. The technology shift from bipolar to CMOS (or BiCMOS) has been apparent in most applications. Even fields like RF, where traditionally GaAs and bipolar were the dominant technologies, now show a trend toward BiCMOS (preferably with a SiGe option) and even plain CMOS for reasons of higher integration and cost reduction. These higher levels of mixed-signal integration, however, also introduce a whole new set of problems and design effects that need to be accounted for in the design process.

Indeed, together with the increase in circuit complexity, the design complexity of today's ICs has increased drastically: 1) due to integration, more and more transistors are combined per IC, performing both analog and digital functions, to be codesigned together with the embedded software; 2) new signal processing algorithms and corresponding system architectures are developed to accommodate new required functionalities and performance requirements (including power) of emerging applications; and 3) due to the rapid evolution of process technologies, the expectation for changing process technology parameters needs to be accounted for in the design cycle. At the same time, many ASIC and ASSP application markets are characterized by shortening product life cycles and tightening time-to-market constraints. The time-to-market factor is very critical for ASICs and ASSPs that eventually end up in consumer, telecom, or computer products: if one misses the initial market window relative to the competition, prices and, therefore, profit can be seriously eroded.

The key to managing this increased design complexity while meeting the shortening time-to-market factor is the **use of computer-aided design** (**CAD**) **and verification tools**. Today's high-speed workstations provide ample power to make large and detailed computations possible. What is needed to expedite the analog and mixed-signal design process is a structured methodology and supporting CAD tools to manage the entire design process and design complexity. CAD tools are also needed to assist or automate many of the routine and repetitive design tasks, taking away the tedium of manually designing these sections and providing the designer with more time to focus on the creative aspects of design. ICs typically are composed of many identical circuit blocks used across different designs. The design of these repetitive blocks can be automated to reduce the design time. In addition, CAD tools can increase the productivity of designers, even for nonrepetitive analog blocks. Therefore, analog CAD and circuit design automation are likely to play a key role in the design process of the next generation of mixed-signal ICs and ASICs. And although the design of mixed-signal ASICs served as the initial impetus for stepping up the efforts in research and development of

analog design automation tools, the technology trend toward integrating complete **systems on a chip** in recent years has provided yet another driving force to bolster analog CAD efforts. In addition, for such systems new design paradigms are being developed that greatly affect how we will design analog blocks. One example is the macrocell design reuse methodology of assembling a system by reusing soft or hard macrocells ("virtual components") that are available on the intellectual property (IP) market and that can easily be mixed and matched in the "silicon board" system if they comply with the virtual socket inferface (VSI) standard [4]. This methodology again poses many new constraints, also on the analog blocks. Platform-based design is another emerging system-level design methodology [5].

In the digital domain, CAD tools are fairly well developed and commercially available today, certainly for the lower levels of the design flow. First, the digital IC market is much larger than the analog IC market. In addition, unlike analog circuits, a digital system can naturally be represented in terms of Boolean representation and programming language constructs, and its functionality can easily be represented in algorithmic form, thus paving the way for a logical transition into automation of many aspects of digital system design. At the present time, many lower-level aspects of the digital design process are fully automated. The hardware is described in a hardware description language (HDL) such as VHDL or Verilog, either at the behavioral level or most often 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 semicustom layout tools (place and route) map this netlist into a correct-by-construction mask-level layout based on a cell library specific for the selected technology process. Research interest is now moving in the direction of system synthesis where a system-level specification is translated into a hardware–software coarchitecture with high-level specifications for the hardware, the software, and the interfaces. Reuse methodologies and platform-based design methodologies are being developed to further reduce the design effort for complex systems. Of course, the level of automation is far from the push-button stage, but the developments are keeping up reasonably well with the chip complexity offered by the technology.

Unfortunately, the story is quite different on the analog side. There are not yet any robust commercial CAD tools to support or automate analog circuit design apart from circuit simulators (in most cases, some flavor of the ubiquitous SPICE simulator [6]) and layout editing environments and their accompanying tools (e.g., some limited optimization capabilities around the simulator, or layout verification tools). Some of the main reasons for this lack of automation are that analog design in general is perceived as less systematic and more heuristic and knowledge-intensive in nature than digital design, and that it has not yet been possible for analog designers to establish a higher level of abstraction that shields all the device-level and process-level details from the higher level design. Analog IC design is a complex endeavor, requiring specialized knowledge and circuit design skills acquired through many years of experience. The variety of circuit schematics and the number of conflicting requirements and corresponding diversity of device sizes is also much larger. In addition, analog circuits are more sensitive to nonidealities and all kinds of higher order effects and parasitic disturbances (crosstalk, substrate noise, supply noise, etc.). These differences from digital design also explain why analog CAD tools cannot simply adapt the digital algorithms, but why specific analog solutions need to be developed that are targeted to the analog design paradigm and complexity. The analog CAD field, therefore, had to evolve on its own, but it turned into a niche field as the analog IC market was smaller than the digital one. As a result, due to the lack of adequate and mature commercial analog CAD tools, analog designs today are still largely being handcrafted with only a SPICE-like simulation shell and an interactive layout environment as supporting facilities. The **design cycle for analog and mixed-signal ICs remains long and error-prone**. Therefore, although analog circuits typically occupy only a small fraction of the total area of mixed-signal ICs, their design is often the bottleneck in mixed-signal systems, both in design time and effort as well as test cost, and they are often responsible for design errors and expensive reruns.

The economic pressure for high-quality yet cheap electronic products and the decreasing time-to-market constraints have clearly **revealed the need in the present microelectronics industry for analog CAD tools** to assist designers with fast and first-time-correct design of analog circuits, or even to automate certain tasks of this design process where possible. The push for more and more integrated systems containing both analog and digital circuitry heavily constrains analog designers. To keep pace with the digital world and to fully exploit the potential offered by the present deep submicrometer VLSI technologies, boosting analog design productivity is a major concern in the industry today. The design time and cost for analog circuits from specification to successful silicon has to be reduced drastically. The risk for design errors impeding first-pass functional (and possibly also parametrically correct) chips has to be eliminated. Second, analog CAD tools can also help to increase the quality of the resulting designs. Before starting detailed circuit implementation, more higher-level explorations and optimizations should be performed at the system architectural level, preferrably across the analog–digital boundary, since decisions at those levels have a much larger impact on key overall system parameters such as power consumption and chip area. Likewise, designs at lower levels should be "automated" where possible. Designers find difficulty in considering multiple conflicting tradeoffs at the same time—computers do not. Computers are adept at trying out and exploring large numbers of competing alternatives. Typical examples are fine-tuning through optimization of an initial handcrafted design and improving design robustness with respect to operating parameter variations (temperature, supply voltage) and/or with respect to manufacturing tolerances and

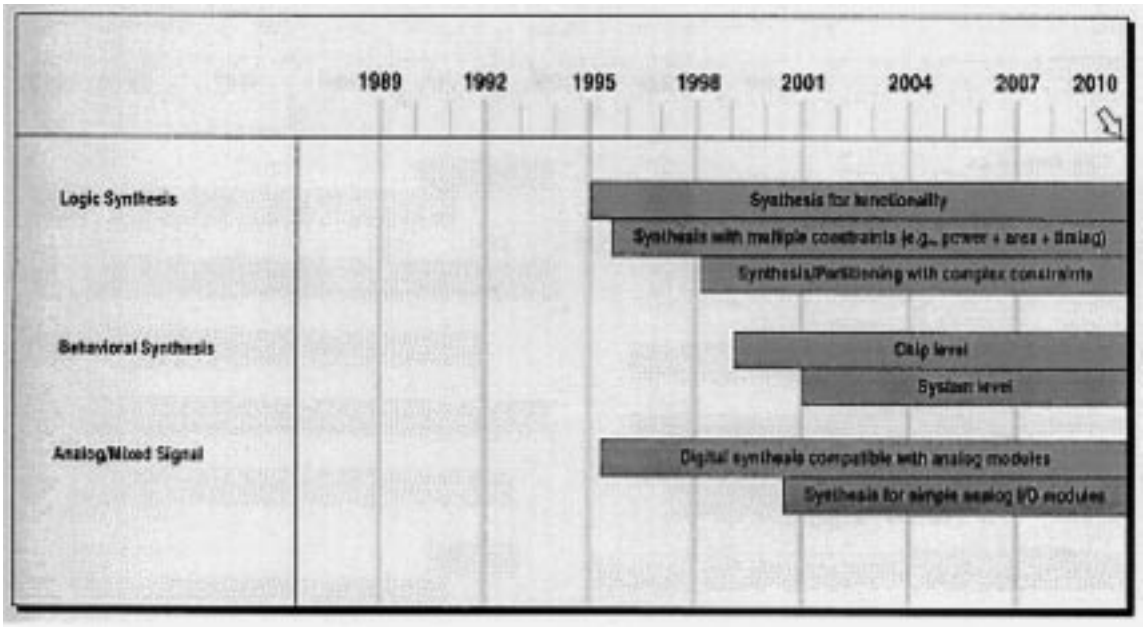

Fig. 2. SIA synthesis potential solutions roadmap [3].

mismatches. Third, the continuous pressure of technology updates and process migrations is a large burden on analog designers. CAD tools could take over a large part of the technology retargeting effort, and could make analog design easier to port or migrate to new technologies. Finally, the SoC design reuse methodology also requires executable models and other information for the analog macrocells to be used in system-level design and verification. Tools and modeling techniques have to be developed to make this possible. This need for analog CAD tools beyond simulation has also clearly been identified in the SIA roadmap, as indicated in Fig. 2, where analog synthesis is predicted to take off somewhere beyond the year 2000 [3].

Despite the lack of commercial analog CAD tools, analog CAD and design automation over the past 15 years has been a field of profound academic and industrial research activity, although with not quite as many researchers as in the digital world, resulting in a slow but steady progress [7]. Some of the aspects of the analog CAD field are fairly mature, some are ready for commercialization, while others are still in the process of exploration and development. The simulation area has been particularly well developed since the advent of the SPICE simulator, which has led to the development of many simulators, including timing simulators in the digital field and the newer generation of mixed-signal and multilevel commercial simulators. Analog circuit and layout synthesis has recently shown promising results at the research level, but commercial solutions are only starting to appear in the marketplace. The development of analog and mixed-signal hardware description languages like VHDL-AMS [8] and Verilog-A/MS [9] is intended to provide a unifying trend that will link the various analog design automation tasks in a coherent framework that supports a more structured analog design methodology from the design conceptualization stage to the manufacturing stage. They also provide a link between the analog and the digital domains, as needed in designing mixed analog–digital ICs and the SoC of the future.

In this survey, the relevant developments to date in analog and mixed-signal CAD will be covered in a general overview. The paper is organized as follows. Section II describes the analog and mixed-signal integrated system design process, as well as a hierarchical design strategy for the analog blocks. Section III then describes general progress and the current status in the various fields of analog CAD: simulation and modeling, symbolic analysis, circuit synthesis and optimization, layout generation, yield analysis and design centering, and test and design for testability. This is illustrated with several examples. Most of the emphasis will be on circuit and layout synthesis as it is key to analog design automation, while other topics such as test will only be covered briefly in this paper. For the sake of completeness, we did not want to omit those topics, but they require overview papers of their own for detailed coverage. Conclusions are then provided in Section IV, and an extensive list of references completes the paper.

II. Analog and Mixed-Signal Design Process

We will now first describe the design flow for mixed-signal integrated systems from concept to chip, followed by the description of a hierarchical design methodology for the analog blocks that can be adopted by analog CAD systems.

A. Mixed-Signal IC Design Flow

Fig. 3 illustrates a possible scenario for the design flow of a complex analog or mixed-signal IC. The various stages that are traversed in the design process are as follows.

1) Conceptual Design: This is typically the product conceptualization stage, where the specifications for a design

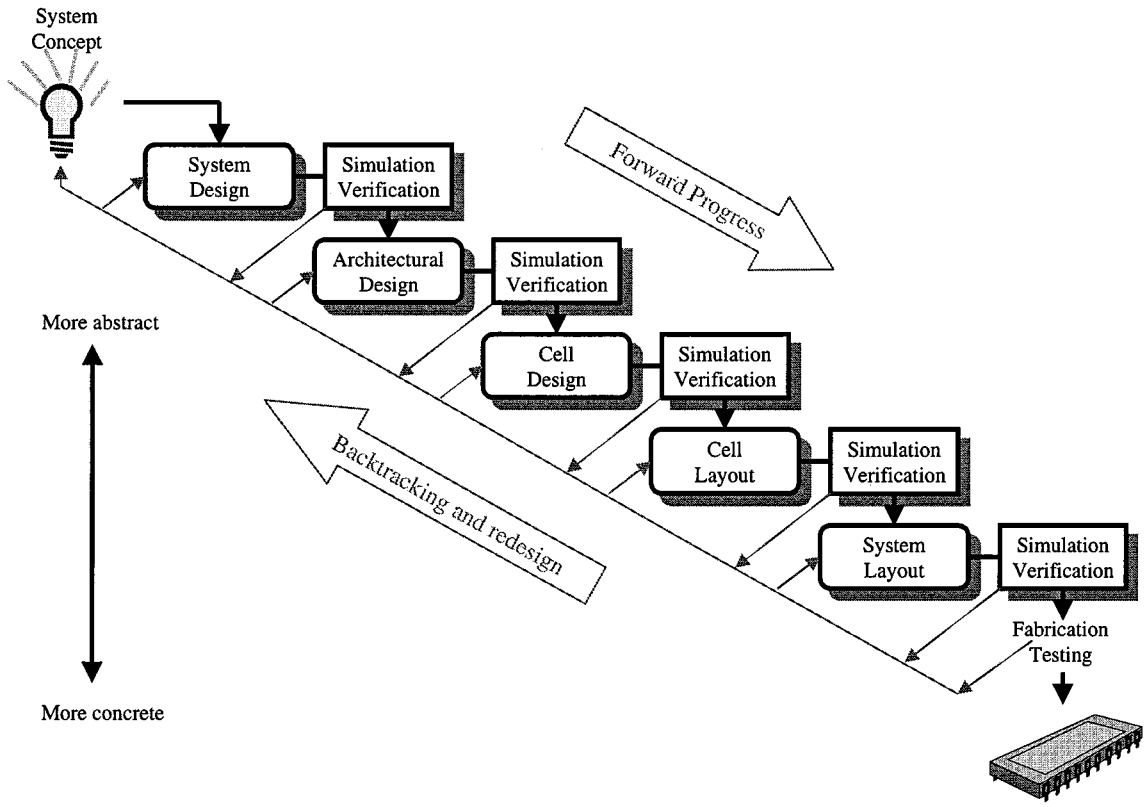

Fig. 3. High-level view of the analog or mixed-signal IC design process.

are gathered and the overall product concept is developed. Careful checking of the specifications is crucial for the later success of the product in its application context. Mathematical tools such as Matlab/Simulink are often used at this stage. This stage also includes setting project management goals such as final product cost and time-to-market, project planning, and tracking.

2) System Design: This is the first stage of the actual design, where the overall architecture of the system is designed and partitioned. Hardware and software parts are defined and both are specified in appropriate languages. The hardware components are described at the behavioral level, and, in addition, the interfaces have to be specified. This stage includes decisions about implementation issues, such as package selection, choice of the target technology, and general test strategy. The system-level partitioning and specifications are then verified using detailed cosimulation techniques.

3) Architectural Design: This stage is the high-level decomposition of the hardware part into an architecture consisting of functional blocks required to realize the specified behavioral description. This includes the partitioning between analog and digital blocks. The specifications of the various blocks that compose the design are defined, and all blocks are described in an appropriate hardware description language (e.g., VHDL and VHDL-AMS). The high-level architecture is then verified against the specifications using behavioral mixed-mode simulations.

4) Cell Design: For the analog blocks, this is the detailed implementation of the different blocks for the given specifications and in the selected technology process, resulting in a fully sized device-level circuit schematic. The stage encompasses both a selection of the proper circuit topology and a dedicated sizing of the circuit parameters. Throughout this process, more complex analog blocks will be further decomposed into a set of subblocks. This whole process will be described in more detail in Section II-B. Manufacturability considerations (tolerances and mismatches) are taken into account in order to guarantee a high yield and/or robustness. The resulting circuit design is then verified against the specifications using SPICE-type circuit simulations.

5) Cell Layout: This stage is the translation of the electrical schematic of the different analog blocks into a geometrical representation in the form of a multilayer layout. This stage involves area optimization to generate layouts that occupy a minimum amount of chip real-estate. The layout is followed by extraction of layout parasitics and detailed circuit-level simulations of the extracted circuit in order to ensure that the performance characteristics do not deviate on account of layout parasitics.

6) System Layout: The generation of the system-level layout of an IC not only includes system-level block place and route, but also power-grid routing. Crosstalk and substrate coupling analysis are important in mixed-signal ICs, and proper measures such as shielding or guarding must also be included. Also, the proper test structures are inserted to make the IC testable. Interconnect parasitics are extracted and detailed verification (e.g., timing analysis) is performed. Finally, the system is verified by cosimulating the hardware part with the embedded software.

7) Fabrication and Testing: This is the processing stage where the masks are generated and the ICs fabricated. Testing is performed during and after fabrication in order to reject defective devices.

Note that any of the many simulation and verification stages throughout this design cycle may detect potential problems with the design failing to meet the target requirements. In that case, **backtracking or redesign** will be needed, as indicated by the upward arrow on the left-hand side of Fig. 3.

B. Hierarchical Analog Design Methodology

This section focuses on the design methodology adopted for the design of analog integrated circuits. These analog circuits could be part of a larger mixed-signal IC. Although at the present time there is no clear-cut general design methodology for analog circuits yet, we outline here the hierarchical design methodology prevalent in many of the emerging experimental analog CAD systems [10]–[15]. For the design of a complex analog macroblock such as a phase-locked loop or an analog-to-digital converter, the analog block is typically decomposed into smaller subblocks (e.g., a comparator or a filter). The specifications of these subblocks are then derived from the initial specifications of the original block, after which each of the subblocks can be designed on its own, possibly by further decomposing it into even smaller subblocks. In this way, constraints are passed down the hierarchy in order to make sure that the top-level block in the end meets the specifications. This whole process is repeated down the decomposition hierarchy until a level is reached that allows a physical implementation (either the transistor level or a higher level in case analog standard cells or IP macrocells are used). The top–down synthesis process is then followed by a bottom–up layout implementation and design verification process. The need for detailed design verification is essential since manufacturing an IC is expensive, and a design needs to be ensured to be fully functional and meet all the design requirements within a window of manufacturing tolerances, before starting the actual fabrication. When the design fails to meet the specifications at some point in the design flow, redesign iterations are needed.

Most experimental analog CAD systems today use a **performance-driven design strategy within such analog design hierarchy**. This strategy consists of the alternation of the following steps in between any two levels i and $i+1$ of the design hierarchy (see Fig. 4):

1) **Top–down path**:
   a) topology selection;
   b) specification translation (or circuit sizing);
   c) design verification.
2) **Bottom–up path**:
   a) layout generation;
   b) detailed design verification (after extraction).

Topology selection is the step of selecting the most appropriate circuit topology that can best meet the given specifications out of a set of already known alternative topologies.

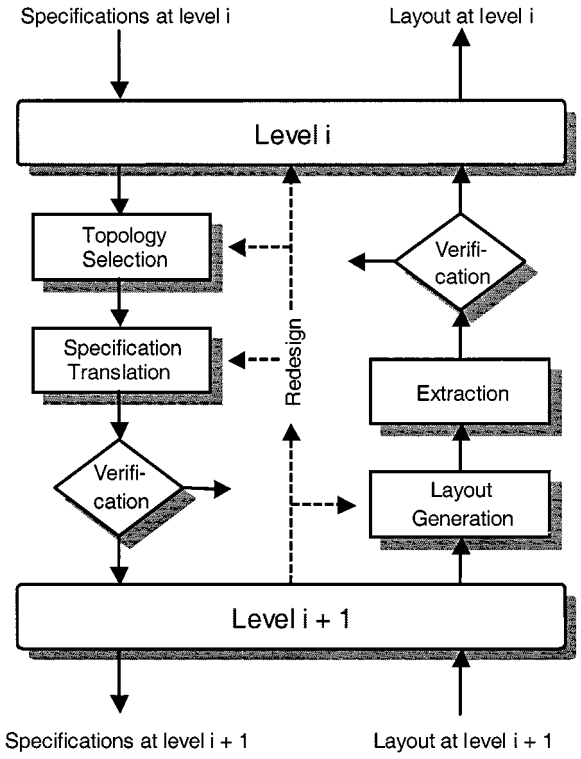

Fig. 4. Hierarchical design strategy for analog circuits.

(An alternative is that the designer develops his/her own new topology.) A topology can be defined hierarchically in terms of lower-level subblocks. For an analog-to-digital converter, for instance, topology selection could be selecting between a flash, a successive approximation, a $\Delta - \Sigma$ or any other topology that can best realize the specifications.

Specification translation is then the step of mapping the specifications for the block under design at a given level (e.g., a converter) into individual specifications for each of the subblocks (e.g., a comparator) within the selected block topology, so that the complete block meets its specifications, while possibly optimizing the design toward some application-specific design objectives (e.g., minimal power consumption). The translated specifications are then verified by means of (behavioral or circuit) simulations before proceeding down in the hierarchy. Behavioral simulations are needed at higher levels in the design hierarchy (when no device-level implementation is available yet); circuit simulations are used at the lowest level in the design hierarchy. At this lowest level, the subblocks are single devices and specification translation reduces to **circuit sizing** (also called circuit dimensioning), which is the determination of all device sizes, element values, and bias parameters in the circuit tuned to the given specifications.

Layout generation is the step of generating the geometrical layout of the block under design, by assembling (place and route) the already generated layouts of the composing subblocks. At the lowest level, these subblocks are individual devices or selected device groupings, which themselves are generated by parameterized procedural device layout generators. Also, power, ground, and substrate connection routing

is part of the layout generation step. This step is followed by extraction and, again, detailed verification and simulation to check the impact of the layout parasitics on the overall circuit performance.

The above methodology is called **performance-driven or constraint-driven**, as the performance specifications are the driving input to each of the steps: each step tries to perform its action (e.g., circuit sizing or layout generation) such that the input constraints are satisfied. This also implies that throughout the design flow, constraints need to be propagated down the hierarchy in order to maintain consistency in the design as it evolves through the various design stages and to make sure that the top-level block in the end meets its target specifications. These propagated constraints may include performance constraints, but also geometrical constraints (for the layout), or manufacturability constraints (for yield), or even test constraints. Design constraint propagation is essential to ensure that specifications are met at each stage of the design, which would also reduce the number of redesign iterations. This is the ultimate advantage of top–down design: **catch problems early in the design flow and, therefore, have a higher chance of first-time success, while obtaining a better overall system design**.

Ideally, one would like to have one clean top–down design path. However, this rarely occurs in practice, as a realistic design needs to account for a number of sometimes hard-to-quantify second-order effects as the design evolves. For instance, a choice of a particular topology for a function block may fail to achieve the required specifications or performance specifications may be too tight to achieve, in which case a redesign step is necessary to alter the block topology or loosen the design specifications. In the above top–down/bottom–up design flow, **redesign or backtracking iterations** may therefore be needed at any point where a design step fails to meet its input specifications. In that case, one or more of the previously executed steps will have to be redone, for example, another circuit topology can be selected instead of the failing one, or another partitioning of subblock specifications can be performed. One of the big differences between analog or mixed-signal designs and the more straight top–down digital designs is exactly the larger number of design iterations needed to come to a good design solution. The adoption of a top–down design methodology is precisely intended to reduce this disadvantage.

A question that can be posed is why the analog circuits need to be redesigned or customized for every new application. The use of a library of carefully selected analog standard cells can be advantageous for certain applications, but is in general inefficient and insufficient. Due to the large variety and range of circuit specifications for different applications, any library will only have a partial coverage for each application, or it will result in an excess power and/or area consumption that may not be acceptable for given applications. Many high-performance applications require an optimal design solution for the analog circuits in terms of power, area, and overall performance. A library-based approach would require an uneconomically large collection of infrequently used cells. Instead, analog circuits are better custom tailored toward each specific application and tools should be available to support this. In addition, the porting of the library cells whenever the process changes is a serious effort, that would also require a set of tools to automate.

The following section in this survey paper will describe the progress and the current state of the art in CAD tool development for the main tasks needed in the above analog design methodology: simulation and modeling, symbolic analysis, circuit synthesis, layout generation, yield estimation and design centering, test, and design for testability.

III. Current Status for the Main Tasks in Analog Design Automation

A. Numerical Simulation of Analog and Mixed-Signal Circuits

A key to ensuring design correctness is the use of simulation tools. Simulation tools have long been in use in the IC design process and provide a quick and cost-effective means of design verification without actual fabrication of the device. The most widely used analog CAD tool today, therefore, is a circuit simulator that numerically calculates the response of the circuit to an input stimulus in the time or frequency domain. In the design methodology of Fig. 4, simulation plays a key role. First of all—and this has been its traditional role—simulation is critical for detailed verification after a design has been completed (before layout as well as after extraction from the layout). Analog integrated circuits are typically impacted by many higher order effects that can severely degrade the circuit performance once fabricated, if the effects are not properly accounted for during the design process. Circuit simulation is a good design aid here by providing the capability of simulating many of the higher order effects and verifying circuit performance prior to fabrication, provided the effects are modeled properly. Second, a result of adopting the top–down design paradigm, simulation is needed to explore tradeoffs and verify designs at a high level, before proceeding with the detailed implementation of the lower-level subblocks. The latter also implies a higher level of modeling for the analog blocks. Finally, executable simulation models are also part of the interface needed to enable the integration of complex systems on a chip by combining IP macrocells.

1) Circuit Simulation: Circuit simulation began with the early development of the **SPICE simulator** [6], [16], and [17], which spawned many of the CAD and IC design efforts and has been the cornerstone of many of today's IC designs. The SPICE simulator is to an analog designer what a calculator is to an engineering school sophomore. Advances in mathematics and the development of many new and efficient numerical algorithms as well as advances in interfaces (e.g., user interfaces, waveform displays, script languages, etc.) have over the years contributed to a vast number of commercial CAD tools. Many variants of the SPICE simulator are now marketed by a number of CAD vendors; many of the IC manufacturers have in-house versions of the SPICE simulator that have been adapted to their own proprietary processes and designs. These simulators have been fine-tuned to

Table 1
Different Analog Hardware Description Levels

	continuous time	discrete time
functional level	signal flow diagram with blocks described by mathematical equations	signal flow diagram with blocks described by mathematical equations
behavioral level	block diagram with building blocks described by differential-algebraic equations and/or s-domain transfer functions	block diagram with sampled-data blocks described by difference-algebraic equations and/or z-domain transfer functions
macro level	circuit with basic elements and equivalent (non)linear controlled sources	circuit with switches, capacitors and equivalent controlled sources
circuit level	circuit with basic elements (transistors, diodes, capacitors, resistors, inductors, etc.)	

meet the convergence criteria of the many difficult-to-simulate ICs. SPICE or its many derivatives have evolved into an established designer utility that is being used both during the design phase (often in a designer-guided trial-and-error fashion) and for extensive postlayout design verification.

A problem that has frustrated analog designers for many years is the limited accuracy of the semiconductor device models used in these simulators, especially for small-signal parameters and on the boundary between different operating regions of the devices (where the earlier models had discontinuities). Fortunately, recent models such as BSIM3 v3, Philips model 9 or EKV look more promising for analog design by providing smooth and continuous transitions between different operating regions [18]. For RF applications, however, even these models are not accurate enough, and the latest research work concentrates on analyzing and modeling the extra effects that become important at higher operating frequencies (e.g., the distributed gate, the resistive bulk, and nonquasi-static effects) [19].

With the explosion of mixed-signal designs, the need has also arisen for simulation tools that allow not only simulation of analog or digital circuits separately, but also simulation of truly mixed analog–digital designs [20]. Simulating the large digital parts with full SPICE accuracy results in very long overall simulation times, whereas efficient event-driven techniques exist to simulate digital circuits at higher abstraction levels than the transistor level. Therefore, mixed-mode simulators were developed that glue together an accurate SPICE-like analog simulator to an efficient digital simulator. These so-called **glued mixed-mode simulators** address the conversions of the signals between analog and digital signal representations and of the appropriate loading effects by inserting interface elements at the boundaries between analog and digital circuitry. Also, the synchronization between the analog kernel with its tiny integration steps and the digital kernel with its events determines the efficiency of the overall simulation. Such synchronization is needed at each time point when an event crosses the boundary between analog or digital. Today, the trend clearly is toward a more unified level of algorithm integration with single-kernel multiple-solver solutions, and commercial solutions following that line have recently appeared in the marketplace.

2) Circuit Modeling: In recent years, the need has also arisen for **higher levels of abstraction to describe and simulate analog circuits**. There are three reasons for this. In a top–down design methodology at higher levels of the design hierarchy, where the detailed lower-level circuit implementations are yet unknown, there is a need for higher-level models describing the pin-to-pin behavior of the circuits rather than the (yet unknown) internal structural implementation. Second, the verification of integrated mixed-signal systems also requires higher description levels for the analog sections, since such integrated systems are computationally too complex to allow a full simulation of the entire mixed-signal design in practical terms. Third, when providing or using analog IP macrocells in a SoC context, the virtual component has to be accompanied by an executable model that efficiently models the pin-to-pin behavior of the virtual component. This model can then be used in system-level design and verification, even without knowing the detailed circuit implementation of the macrocell.

To solve those three problems, modeling paradigms and languages from the digital world have migrated to the analog domain. For this reason, macro, behavioral, and functional simulation levels have been developed for analog circuits besides the well-known circuit level [21]. For a commercial simulator to be useful in current industrial mixed-signal design practice, it therefore has to be capable of simulating a system containing a mix of analog blocks described at different levels and in different domains, in combination with digital blocks. This requires a true mixed-signal, multilevel, mixed-domain simulator.

Table 1 gives an overview of the different analog description levels, both for continuous-time and discrete-time analog circuits [21]. In a *macromodel*, an equivalent but computationally cheaper circuit representation is used that has approximately the same behavior as the original

circuit. Equivalent sources combine the effect of several other elements that are eliminated from the netlist. The simulation speed-up is roughly proportional to the number of nonlinear devices that can be eliminated. In a *behavioral or functional model*, a purely mathematical description of the input–output behavior of the block is used. This typically will be in the form of a set of differential-algebraic equations (DAE) and/or transfer functions. At the behavioral level, conservation laws still have to be satisfied on the pins connecting different blocks. At the functional level, this is no longer the case and the simulated system turns into a kind of signal-flow diagram. Fig. 5 shows an example of the output response of a CMOS current-steering digital-to-analog converter, modeled at the full device level Fig. 5(b) and at the behavioral level Fig. 5(a). The responses are quite similar (the error between the two time-domain responses for the same input signal is less than 1%), while the behavioral model simulates about 1000 times faster.

To allow an easy exchange of these models across different simulators and different users, the need arose for **standardized analog hardware description languages** in which to describe these higher-level models. These language standards have to provide a consistent way of representing and sharing design information across the different design tasks and across the design hierarchy, and, therefore, provide a unifying trend to link the various tools in a global analog CAD system. For mixed-signal designs, the analog HDLs had to be compatible with the existing digital HDLs (such as VHDL and Verilog). Several parallel analog or mixed-signal HDL language standardization efforts, therefore, have been initiated, recently resulting in the standardized languages VHDL-AMS [8] and Verilog-A/MS [9]. The VHDL-AMS language targets the mixed-signal domain and is a superset of the digital VHDL language. Verilog-A for the analog part and Verilog-MS for the mixed-signal part target compatibility with the Verilog language. Recently, also, the standardization of an extension of VDHL-AMS toward RF has been started.

One of the remaining difficulties with higher-level analog modeling is the automatic characterization of analog circuits and more particularly the **automatic generation of analog macromodels or behavioral models from a given design**. This is a difficult problem area that needs to be addressed in the near future, as it might turn out to be the biggest hurdle for the adoption of these high-level modeling methodologies and AHDLs in industrial design practice. Current approaches can roughly be divided into fitting approaches and constructive approaches. In the fitting approaches, a parameterized model (for example, a rational transfer function, a more general set of equations, or even a neural network model) is first proposed by the model developer and the values of the parameters are then fitted by some least-square error optimization so that the model response matches as closely as possible the response of the real circuit [22]–[24]. The problem with this approach is that first a good model template must be proposed. The second class of methods, therefore, tries to generate or build a model from the underlying circuit description. One approach, for instance, uses symbolic simplification techniques to simplify the physical equations that describe the circuit up to a maximum error bound [25]. Up until now, however, the gains in CPU time were not high enough for practical circuits. More research in this area is definitely needed.

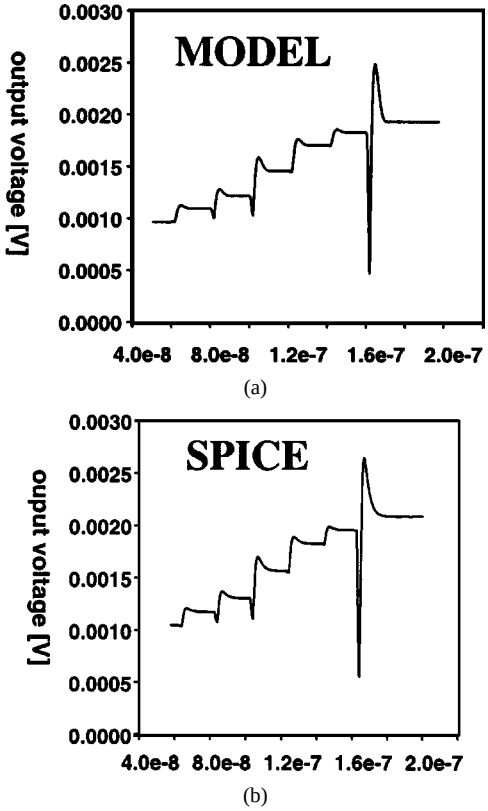

Fig. 5. Comparison of the output response to the same input waveform of a digital-to-analog converter modeled at the behavioral level (a) and at the circuit level (b). The horizontal axis is time in seconds.

3) Dedicated Simulation Techniques: In addition to the above general-purpose simulation tools for analog and mixed-signal circuits, other techniques or tools have been developed for dedicated purposes. An important class of circuits that are used in many signal processing and communication systems are the switched circuits, like switched-capacitor and, more recently, switched-current circuits. Their switched nature, with the resulting switching transients, requires many small numerical integration steps to be simulated within each clock phase if a standard SPICE simulator is used. On the other hand, advantage can be taken of the periodically switched nature of the circuits and the fact that in a time-discrete circuit the signals are only important and, thus, only have to be calculated at specific time points (e.g., the end of each clock phase). This is exploited in several switched-capacitor simulation tools like SWITCAP [26], [27] and SWAP [28] but also in dedicated tools like TOSCA that analyzes switched-capacitor-based $\Delta - \Sigma$ converters [29].

Another important domain is **RF simulation**, needed for instance when developing circuits for wireless applications, where modulated signals have to be simulated and effects like noise, distortion, and intermodulation become impor-

tant. Here, techniques have been developed to directly simulate the steady-state behavior of these circuits without having to wait for the decay of the initial transients [30], [31]. In the time domain, shooting methods are used for this, which tend to be more suited for strongly nonlinear circuits. In the frequency domain, harmonic balance methods are used, which allow a simulation of the steady-state behavior of nonlinear circuits driven by one- or two-tone signals but which historically required large CPU times and memory sizes for large circuits or for strong nonlinearities. Recently, the implicit matrix technique in combination with both shooting or harmonic balance methods extended the range of these methods to much larger circuits [32]. In parallel, other techniques have been developed such as the envelope simulation technique [33], which combines time and frequency domain simulation to efficiently calculate the circuit's response to truly modulated signals by separating the carrier from the modulation signal. Other dedicated simulation algorithms have been developed for specific applications such as the high-level analysis of entire analog RF receiver front ends in the ORCA tool [34], or for the analysis of nonlinear noise and phase noise in both autonomous and driven circuits such as oscillators, mixers, and frequency synthesizers [35], [36].

An important problem in deep submicrometer technologies where interconnect delays are exceeding gate delays is the **analysis of interconnect networks** during postlayout timing verification. Accurate models for each wire segment and the driving gates are needed, which makes the overall interconnect network too complex to simulate. Therefore, recent developments try to improve the efficiency of timing verification while keeping the accuracy by using piecewise-linear models for gates and model-order reduction techniques for the interconnect network [37]. The complexity of the interconnect network can be reduced by techniques such as asymptotic waveform evaluation (AWE) [38] or related variants such as Padé via Lanczos (PVL), that use moment matching and Padé approximation to generate a lower order model for the response of a large linear circuit like an interconnect network. The early AWE efforts used explicit moment matching techniques, which could generate unstable reduced-order models. Subsequent developments using iterative methods resulted in methods like PVL that overcome many of the deficiencies of the earlier AWE efforts, and stability is now guaranteed using techniques like Arnoldi transformations [39]. The interconnect delay problem has become so important that it is now driving the layout generation to get in-time timing closure, and that it even is becoming essential for synthesis (where, of course, estimation techniques must be used) [40].

An important problem in mixed-signal ICs is **signal integrity analysis**: *the analysis of crosstalk and couplings such as capacitive or inductive interconnect couplings or couplings through the supply lines or the substrate*. Crosstalk can be a limiting factor in today's high-speed circuits with many layers of interconnect. Substrate or supply coupling noise is particularly important for analog circuits, especially where they have to sense small input signals, such as in receiver front ends. Research has been going on to find efficient yet accurate techniques to analyze these problems, which depend on the geometrical configuration and, therefore, are in essence three-dimensional field-solving problems. Typically, finite difference methods or boundary element methods are used to solve for the substrate potential distribution due to injected noise sources [41]–[44]. Recently, these methods have been speeded up with similar acceleration techniques as in RF or interconnect simulation, e.g., using an eigendecomposition technique [45]. Their efficiency even allows one to perform some substrate design optimizations [46]. A problem is that the noise-generating sources (i.e., the switching noise injected by the digital circuitry) are not accurately known, but vary with time depending on the input signals or the embedded programs, and, therefore, have to be estimated statistically. Some attempts to solve this problem characterize every cell in a digital standard cell library by the current they inject in the substrate due to an input transition, and then calculate the total injection of a complex system by summing the contributions of all switching cells over time [47].

B. Symbolic Analysis of Analog Circuits

Analog design is a very complex and knowledge-intensive process, which heavily relies on circuit understanding and related design heuristics. Symbolic circuit analysis techniques have been developed to help designers gain a better understanding of a circuit's behavior. A symbolic simulator is a computer tool that takes as input an ordinary (SPICE-type) netlist and returns as output (simplified) analytic expressions for the requested circuit network functions in terms of the symbolic representations of the frequency variable and (some or all of) the circuit elements [48], [49]. They perform the same function that designers traditionally do by hand analysis (even the simplification). The difference is that the analysis is now done by the computer, which is much faster, can handle more complex circuits, and does not make as many errors. An example of a complicated BiCMOS opamp is shown in Fig. 6. The (simplified) analytic expression for the differential small-signal gain of this opamp has been analyzed with the SYMBA tool [50] and is shown below

$$A_{V0} = \frac{g_{m,M2}}{g_{m,M1}} \frac{g_{m,M4}}{\left(\frac{g_{o,M4}g_{o,M5}}{g_{m,M5}+g_{mb,M5}} + \frac{G_a+g_{o,M9}+g_{o,Q2}}{\beta_{Q2}} \right)}.$$

The symbolic expression gives a better insight into which small-signal circuit parameters predominantly determine the gain in this opamp and how the user has to design the circuit to meet a certain gain constraint. In this way, symbolic circuit analysis is complementary to numerical (SPICE) circuit simulation, which was described in the previous section. Symbolic analysis provides a different perspective that is more suited for obtaining insight in a circuit's behavior and for circuit explorations, whereas numerical simulation is more appropriate for detailed design validation once a design point has been decided upon. In addition, the generated symbolic

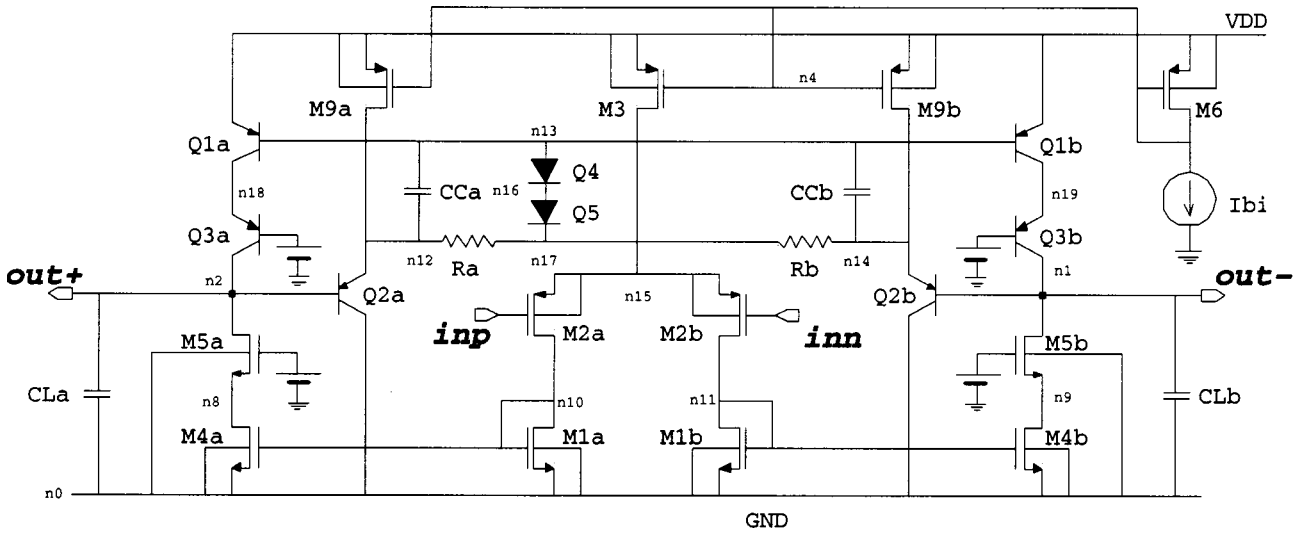

Fig. 6. BiCMOS operational amplifier to illustrate symbolic analysis.

design equations also constitute a model of the circuit's behavior that can be used in CAD tasks such as analog synthesis, statistical analysis, behavioral model generation, or formal verification [48].

At this moment, only symbolic analysis of linear or small-signal linearized circuits in the frequency domain is possible, both for continuous-time and discrete-time (switched) analog circuits [48], [49], [51]. In this way, symbolic expressions can be generated for transfer functions, impedances, noise functions, etc. In addition to understanding the first-order functional behavior of an analog circuit, a good understanding of the second-order effects in a circuit is equally important for the correct functioning of the design in its system application later on. Typical examples are the PSRR and the CMRR of a circuit, which are limited by the mismatches between circuit elements. These mismatches are represented symbolically in the formulas. Another example is the distortion or intermodulation behavior, which is critical in telecom applications. The technique of symbolic simulation has been extended to the symbolic analysis of distortion and intermodulation in weakly nonlinear analog circuits where the nonlinearity coefficients of the device small-signal elements appear in the expressions [52].

Exact symbolic solutions for network functions, however, are too complex for linear(ized) circuits of practical size, and even impossible to calculate for many nonlinear effects. Even rather small circuits lead to an astronomically high number of terms in the expressions, that can neither be handled by the computer nor interpreted by the circuit designer. Therefore, since the late 1980s, and in principle similar to what designers do during hand calculations, dedicated symbolic analysis tools have been developed that use heuristic simplification and pruning algorithms based on the relative importance of the different circuit elements to reduce the complexity of the resulting expressions and retain only the dominant contributions within user-controlled error tolerances. Examples of such tools are ISAAC [51], SYNAP [53], and ASAP [54] among many others. Although successful for relatively small circuits, the fast increase of the CPU time with the circuit size restricted their applicability to circuits between 10 and 15 transistors only, which was too small for many practical applications.

In recent years, however, an algorithmic breakthrough in the field of symbolic circuit analysis has been realized. The techniques of simplification before and during the symbolic expression generation, as implemented in tools like SYMBA [50] and RAINIER [55], highly reduce the computation time and, therefore, enable the symbolic analysis of large analog circuits of practical size (like the entire 741 opamp or the example of Fig. 6). In simplification before generation (SBG), the circuit schematic, or some associated matrix or graph(s), are simplified before the symbolic analysis starts [56], [57]. In simplification during generation (SDG), instead of generating the exact symbolic expression followed by pruning the unimportant contributions, the desired simplified expression is built up directly by generating the contributing dominant terms one by one in decreasing order of magnitude, until the expression has been generated with the desired accuracy [50], [55].

All these techniques, however, still result in large, expanded expressions, which restricts their usefulness for larger circuits. Therefore, for really large circuits, the technique of hierarchical decomposition has been developed [58], [59]. The circuit is recursively decomposed into loosely connected subcircuits. The lowest-level subcircuits are analyzed separately and the resulting symbolic expressions are combined according to the decomposition hierarchy. This results in the global nested expression for the complete circuit, which is much more compact than the expanded expression. The CPU time increases about linearly with the circuit size, provided that the coupling between the different subcircuits is not too strong. Another compact representation of symbolic expressions was presented recently. Following the use of binary decision diagrams in logic synthesis, determinant decision diagrams (DDD) have been proposed as a technique to canonically represent determinants in a compact nested format [60]. The advantage is that all operations on these DDDs are linear with the size of the DDD,

but the DDD itself is not always linear with the size of the circuit. This technique has been combined with hierarchical analysis in [61]. Further investigation will have to prove the usefulness of this technique in practice.

Based on the many research results in this area over the last decade, it can be expected that symbolic analysis techniques will soon emerge in the commercial EDA marketplace and that they will soon be part of the standard tool suite of every analog designer. In the meantime, new (possibly heuristic) algorithms for the symbolic analysis of transient and large-signal circuit characteristics are currently being developed in academia.

C. Analog Circuit Synthesis and Optimization

The first step in the analog design flow of Fig. 4 is analog circuit synthesis, which consists of two tasks: topology selection and specification translation. Synthesis is a critical step since most analog designs require a custom optimized design and the number of (often conflicting) performance requirements to be taken into account is large. Analog circuit synthesis is the inverse operation of circuit analysis. During analysis, the circuit topology and the subblock parameters (such as device sizes and bias values) are given and the resulting performance of the overall block is calculated, as is done in the SPICE simulator. During synthesis, on the other hand, the block performance is specified and an appropriate topology to implement this block has to be decided first. This step is called topology selection. Subsequently, values for the subblock parameters have to be determined, so that the final block meets the specified performance constraints. This step is called specification translation at higher levels in the design hierarchy, in which case performance specifications of subblocks have to be determined, or circuit sizing at the device level, in which case the sizes and biasing of all devices have to be determined. See Fig. 7 for an illustration of this flow for low-level cells. The inversion process inherent to synthesis, however, is not a one-to-one mapping, but typically is an underconstrained problem with many degrees of freedom. *The different analog circuit synthesis systems that have been explored up till now can be classified based on how they perform topology selection and how they eliminate the degrees of freedom during specification translation or circuit sizing*. In many cases, the initial sizing produces a near optimal design that is further fine-tuned with a circuit optimization tool. The performance of the resulting design is then verified using detailed circuit simulations with a simulator such as SPICE, and when needed the synthesis process is iterated to arrive at a close-fit design. We will now discuss the two basic steps in more detail.

1) Topology Selection: Given a set of performance specifications and a technology process, a designer or a synthesis tool must first select a circuit schematic that is most suitable to meet the specifications at minimal implementation cost (power, chip area). This problem can be solved by selecting a schematic from among a known set of alternative topologies such as stored in a library (topology selection), or by generating a new schematic, for example by modifying an existing schematic. Although the earliest synthesis approaches

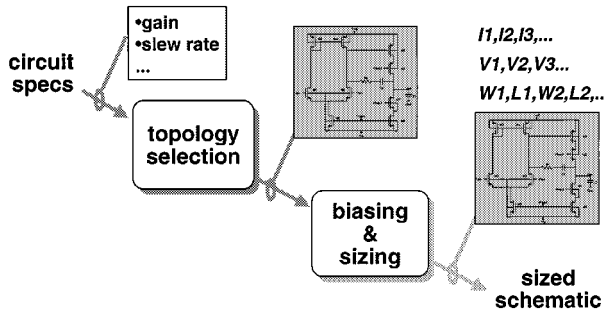

Fig. 7. Basic flow of analog circuit synthesis for a basic cell: topology selection and circuit sizing.

considered topology selection and sizing together, the task of topology selection has received less attention in recent years, where the focus was primarily on the circuit sizing. Finding the optimal circuit topology for a given set of performance specifications is rather heuristic in nature and brings to bear the real expert knowledge of a designer. Thus, it was only natural that the first topology selection approaches like in OASYS [10], BLADES [62], or OPASYN [63] were rather heuristic in nature in that they used rules in one format or another to select a proper topology (possibly hierarchically) out of a predefined set of alternatives stored in the tool's library.

Later approaches worked in a more quantitative way in that they calculate the feasible performance space of each topology that fits the structural requirements, and then compare that feasible space to the actual input specifications during synthesis to decide on the appropriateness and the ordering of each topology. This can for instance be done using interval analysis techniques [64] or using interpolation techniques in combination with adaptive sampling [65]. In all these programs, however, topology selection is a separate step. There are also a number of optimization-based approaches that integrate topology selection with circuit sizing as part of one overall optimization loop. This was done using a mixed integer-nonlinear programming formulation with Boolean variables representing topological choices [66], or by using a nested simulated evolution/annealing loop where the evolution algorithm looks for the best topology and the annealing algorithm for the corresponding optimum device sizes [67]. Another approach that uses a genetic algorithm to find the best topology choice was presented in DARWIN [68]. Of these methods, the quantitative and optimization-based approaches are the more promising developments that address the topology selection task in a deterministic fashion as compared to the rather ad-hoc heuristic methods.

2) Analog Circuit Sizing: Once an appropriate topology has been selected, the next step is specification translation, where the performance parameters of the subblocks in the selected topology are determined based on the specifications of the overall block. At the lowest level in the design hierarchy, this reduces to circuit sizing where the sizes and biasing of all devices have to be determined such that the final circuit meets the specified performance constraints. This mapping from performance specifications into proper, preferrably optimal, device sizes and biasing for a selected analog circuit topology

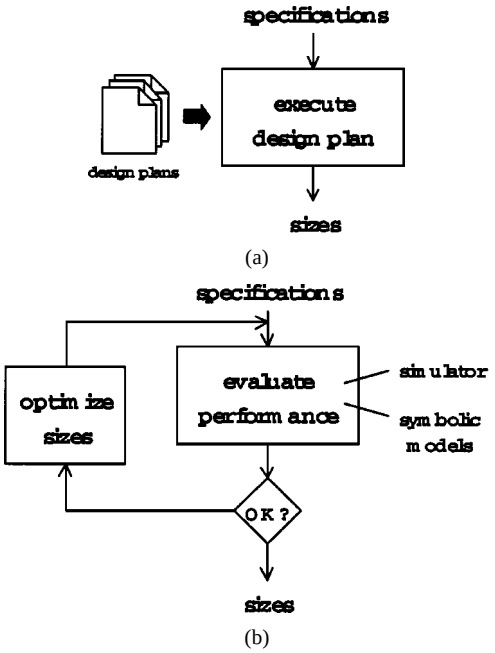

Fig. 8. The two basic approaches toward analog circuit synthesis: (a) the knowledge-based approach using procedural design plans, and (b) the optimization-based approach.

in general involves solving the set of physical equations that relate the device sizes to the electrical performance parameters. However, solving these equations explicitly is in general not possible, and analog circuit sizing typically results in an underconstrained problem with many degrees of freedom. The two basic ways to solve for these degrees of freedom in the analog sizing process are either by exploiting analog design knowledge and heuristics, or by using optimization techniques. These two basic methods, which are schematically depicted in Fig. 8, correspond to the two broad classes of approaches adopted toward analog circuit synthesis, i.e., the knowledge-based approaches and the optimization-based approaches [7], [69].

a) Knowledge-Based Analog Sizing Approaches: The first generation of analog circuit synthesis systems presented in the mid to late 1980s were **knowledge-based**. Specific heuristic design knowledge about the circuit topology under design (including the design equations but also the design strategy) was acquired and encoded explicitly in some computer-executable form, which was then executed during the synthesis run for a given set of input specifications to directly obtain the design solution. This approach is schematically illustrated in Fig. 8(a). The knowledge was encoded in different ways in different systems.

The IDAC tool [70] used manually derived and prearranged design plans or design scripts to carry out the circuit sizing. The design equations specific for a particular circuit topology had to be derived and the degrees of freedom in the design had to be solved explicitly during the development of the design plan using simplifications and design heuristics. Once the topology was chosen by the designer, the design plan was loaded from the library and executed to produce a first-cut design that could further be fine-tuned through local optimization. The big advantage of using design plans is their fast execution speed, which allows for fast-performance space explorations. The approach also attempts to take advantage of the knowledge of analog designers. IDAC's schematic library was also quite extensive, and it included various analog circuits such as voltage references, comparators, etc., besides operational amplifiers. The big disadvantages of the approach are the lack of flexibility in the hardcoded design plans and the large time needed to acquire the design equations and to develop a design plan for each topology and design target, as analog design heuristics are very difficult to formalize in a general and context-independent way. It has been reported [71] that the creation of a design script or plan typically took four times more effort than is needed to actually design the circuit once. A given topology must therefore at least be used in four different designs before it is profitable to develop the corresponding design plan. Considering the large number of circuit schematics in use in industrial practice, this large setup time essentially restricted the commercial usability of the IDAC tool and limited its capabilities to the initial set of schematics delivered by the tool developer. Also, the integration of the tool in a spreadsheet environment under the name PlanFrame [72] did not fundamentally change this. Note that due to its short execution times, IDAC was intended as an interactive tool: the user had to choose the topology him/herself and also had to specify values for the remaining degrees of freedom left open in the design plan.

OASYS [10] adopted a similar design-plan-based sizing approach where every (sub)block in the library had its own handcrafted design plan, but the tool explicitly introduced hierarchy by representing topologies as an interconnection of subblocks. For example, a circuit like an opamp was decomposed into subcircuits like a differential pair, current mirrors, etc., and not represented as one big device-level schematic as in IDAC. OASYS also added a heuristic approach toward topology selection, as well as a backtracking mechanism to recover from design failures. As shown in Fig. 9, the complete flow of the tool was then an alteration of topology selection and specification translation (the latter by executing the design plan associated with the topology) down the hierarchy until the device level is reached. If the design does not match the desired performance characteristics at any stage in this process, OASYS backtracks up the hierarchy, trying alternate configurations for the subblocks. The explicit use of hierarchy allowed to reuse design plans of lower-level cells while building up higher-level-cell design plans and, therefore, also leveraged the number of device-level schematics covered by one top-level topology template. Although the tool was used successfully for some classes of opamps, comparators and even a data converter, collecting and ordering all the design knowledge in the design plans still remained a huge manual and time-consuming job, restricting the practical usefulness of the tool. The approach was later adopted in the commercial MIDAS system [71], which was used successfully in-house for certain types of data converters. Also, AZTECA [73] and CATALYST [74] use the design-plan approach for the high-level design of successive-approximation and high-speed CMOS data converters, respectively. Inspired

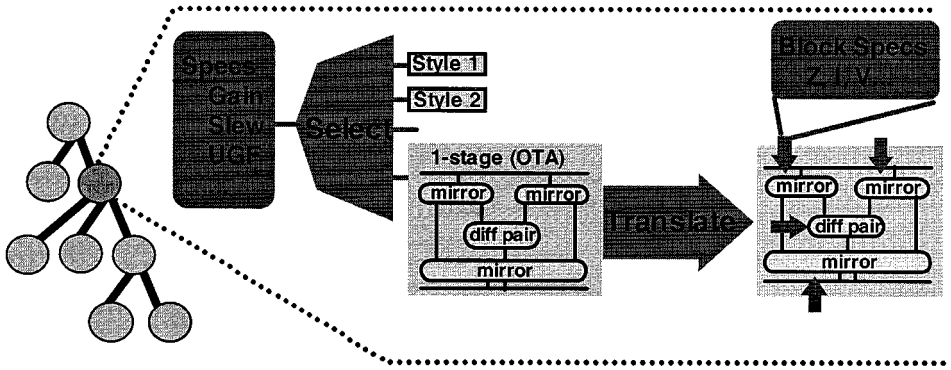

Fig. 9. Hierarchical alternation of topology selection and specification translation down the design hierarchy.

by artificial intelligence research, also other ways to encode the knowledge have been explored, such as in BLADES [62], which is a rule-based system to size analog circuits, in ISAID [75], [76] or in [77].

In all these methods, the heuristic design knowledge of an analog designer turned out to be difficult to acquire and to formalize explicitly, and the manual acquisition process was very time consuming. In addition to analytic equation-based design knowledge, procedural design knowledge is also required to generate design plans, as well as specialized knowledge to support tasks such as failure handling and backtracking. The overhead to generate all this was too large compared to a direct design of the circuit, restricting the tools basically to those circuits that were delivered by the tool developers. Their coverage range was found to be too small for the real-life industrial practice and, therefore, these first approaches failed in the commercial marketplace.

b) Optimization-Based Analog Sizing Approaches: In order to make analog synthesis systems more flexible and extendible for new circuit schematics, an alternative approach was followed starting in the late 1980s. This research resulted in a second generation of methods, the optimization-based approaches. These use numerical optimization techniques to implicitly solve for the degrees of freedom in analog design while optimizing the performance of the circuit under the given specification constraints. These strategies also strive to automate the generation of the required design knowledge as much as possible, e.g., by using symbolic analysis techniques to automatically derive many of the design equations and the sizing plans, or to minimize the explicitly required design knowledge by adopting a more equation-free simulation-oriented approach. This optimization-based approach is schematically illustrated in Fig. 8(b). At each iteration of the optimization routine, the performance of the circuit has to be evaluated. Depending on which method is used for this performance evaluation, two different subcategories of methods can be distinguished.

In the *subcategory of equation-based optimization approaches*, (simplified) analytic design equations are used to describe the circuit performance. In approaches like OPASYN [63] and STAIC [78], the design equations still had to be derived and ordered by hand, but the degrees of freedom were resolved implicitly by optimization. The OPTIMAN tool [79] added the use of a global simulated annealing algorithm, but also tried to solve two remaining problems. Symbolic simulation techniques were developed to automate the derivation of the (simplified) analytic design equations needed to evaluate the circuit performance at every iteration of the optimization [69]. Today, the ac behavior (both linear and weakly nonlinear) of relatively large circuits can already be generated automatically. The second problem is then the subsequent ordering of the design equations into an application-specific design or evaluation plan. Also, this step was automated using constraint programming techniques in the DONALD tool [80]. Together with a separate topology-selection tool based on boundary checking and interval analysis [64] and a performance-driven layout generation tool [81], all these tools are now integrated into the AMGIE analog circuit synthesis system [82] that covers the complete design flow from specifications over topology selection and circuit sizing down to layout generation and automatic verification. An example of a circuit that has been synthesized with this AMGIE system is the particle/radiation detector front end of Fig. 10, which consists of a charge-sensitive amplifier (CSA) followed by an n-stage pulse-shaping amplifier (PSA). All opamps are complete circuit-level schematics in the actual design as indicated in the figure. A comparison between the specifications and the performances obtained by an earlier manual design of an expert designer and by the fully computer-synthesized circuit is given in Table 2. In the experiment, a reduction of the power consumption with a factor of 6 (from 40 to 7 mW) was obtained by the synthesis system compared to the manual solution. Also, the final area is slightly smaller. The layout generated for this example is shown in Fig. 11.

The technique of equation-based optimization has also been applied to the high-level synthesis of $\Delta-\Sigma$ modulators in the SD-OPT tool [83]. The converter architecture is described by means of symbolic equations, which are then used in a simulated-annealing-like optimization loop to derive the optimal subblock specifications from the specifications of the converter. Recently, a first attempt was presented toward the full behavioral synthesis of analog systems from an (annotated) VHDL-AMS behavioral description. The VASE tool follows a hierarchical two-layered optimization-based design-space exploration approach to

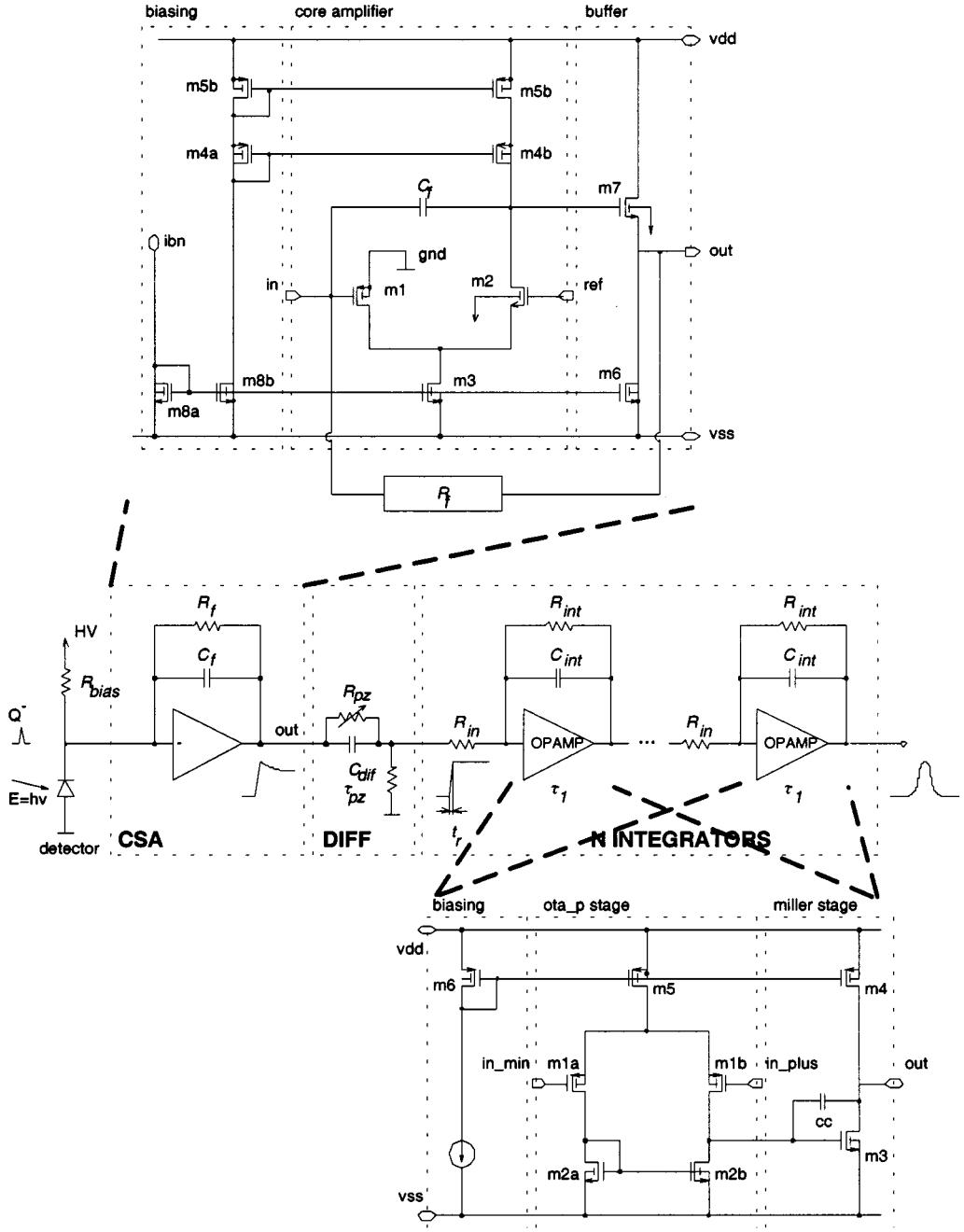

Fig. 10. Particle/radiation detector front end as example for analog circuit synthesis. (The opamp and filter stage symbols represent full circuit schematics as indicated.)

produce sized subblocks from behavioral specifications [84]. A branch-and-bound algorithm with efficient solution-space pruning first generates alternative system topologies by mapping the specifications via a signal-flow-graph representation onto library elements. For each resulting topology a genetic-algorithm-based heuristic method is then executed for constraint transformation and subblock synthesis, which concurrently transforms system-level constraints into subblock design parameters (e.g., a bias current) and fixes the topologies and transistor sizes for all subblocks. For reasons of efficiency the performances at all levels in the considered hierarchy are estimated using analytic equations relating design parameters to performance characteristics. The genetic algorithms operating at the different levels are speeded up by switching from traditional genetic operators to directed-interval-based operators that rely on characterization tables with qualitative sensitivity information to help focusing the search process in promising local regions.

In general, the big advantages of these analytic approaches are their fast evaluation time and their flexibility in manipulation possibilities. The latter is reflected in the freedom to choose the independent input variables, which has a large impact on the overall evaluation efficiency [85], as well as the possibility to perform more symbolic manipulations. Recently, it has been shown that the design of CMOS opamps can be formulated (more precisely, it can be fairly well ap-

Table 2
Example of Analog Circuit Synthesis Experiment with the AMGIE System

performance	specification	manual design	automated synthesis
peaking time	< 1.5 ms	1.1 ms	1.1 ms
counting rate	> 200 kHz	200 kHz	294 kHz
noise	< 1000 RMS e-	750 RMS e-	905 RMS e-
gain	20 V/fC	20 V/fC	21 V/fC
output range	> -1..1 V	-1..1 V	-1.5..1.5 V
power	minimal	40 mW	7 mW
area	minimal	0.7 mm^2	0.6 mm^2

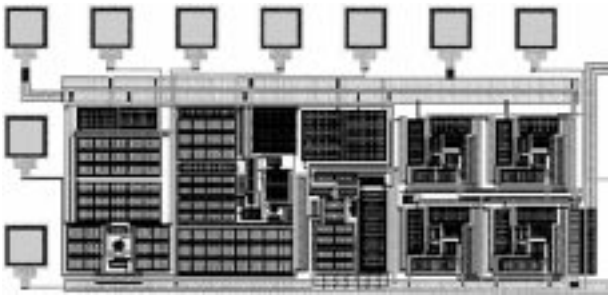

Fig. 11. Layout of the particle/radiation detector front end generated with the AMGIE analog synthesis system.

proximated) as a posynomial convex optimization problem that can then be solved using geometric programming techniques, producing a close-by first-cut design in an extremely efficient way [86]. The initial optimization time literally reduces to seconds. The same approach has been applied to some other circuits as well [87]. The big drawback of the analytic methods, however, is that the design equations still have to be derived and, despite the progress in symbolic circuit analysis, not all design characteristics (such as transient or large-signal responses) are easy to capture in analytic equations with sufficient accuracy. For such characteristics either rough approximations have to be used, which undermines the sense of the whole approach, or one has to fall back on numerical simulations. This problem has sparked research efforts to try to develop equation-free approaches.

Therefore, in recent years and with improving computer power, a second *subcategory of simulation-based optimization approaches* toward analog circuit synthesis has been developed. These methods perform some form of full numerical simulation to evaluate the circuit's performance in the inner loop of the optimization [see Fig. 8(b)]. Although the idea of optimization-based design for analog circuits dates back at least 30 years [88], it is only recently that the computer power and numerical algorithms have advanced far enough to make this really practical. For a limited set of parameters circuit optimization was already possible in DELIGHT.SPICE [89]. This method is most favorable in fine-tuning an already designed circuit to better meet the specifications, but the challenge in automated synthesis is to solve for all degrees of freedom when no good initial starting point can be provided. To this end, the FRIDGE tool [90] calls a plain-vanilla SPICE simulation at every iteration of a simulated-annealing-like global optimization algorithm. In this way, it is able to synthesize low-level analog circuits (e.g., opamps) with full SPICE accuracy. Performance specifications are divided in design objectives and strong and weak constraints. The number of required simulations is reduced as much as possible by adopting a fast cooling schedule with reheating to recover from local minima. Nevertheless, many simulations are performed, and the number of optimization parameters and their range has to be restricted in advance by the designer. The introduction of a new circuit schematic in such an approach is relatively easy, but the drawback remains the long run times, especially if the initial search space is large.

An in-between solution was therefore explored in the ASTRX/OBLX tool [91] where the simulation itself is speeded up by analyzing the linear (small-signal) characteristics more efficiently than in SPICE by using Asymptotic Waveform Evaluation [38]. For all other characteristics equations still have to be provided by the designer. So this is essentially a mixed equation-simulation approach. The ASTRX subtool compiles the initial synthesis specification into an executable cost function. The OBLX subtool then numerically searches for the minimum of this cost function via simulated annealing, hence determining the optimal circuit sizing. To achieve accuracy in the solution, encapsulated industry-standard models are used for the MOS transistors. For efficiency the tool also uses a dc-free biasing formulation of the analog design problem, where the dc constraints [i.e., Kirchhoff current law (KCL) at every node] are not imposed by construction at each optimization iteration, but are solved by relaxation throughout the optimization run by adding the KCL violations as penalty terms to the cost function. At the final optimal solution, all the penalty terms are driven to zero, thus resulting in a KCL-correct and thus electrically consistent circuit in the end. ASTRX/OBLX has been applied successfully to a wide variety of cell-level designs, such as a $2\times$ gain stage [92], but the CPU times remain large. The tool is also most suited only when the circuit behavior is relatively linear, because the other characteristics still require equations to be derived.

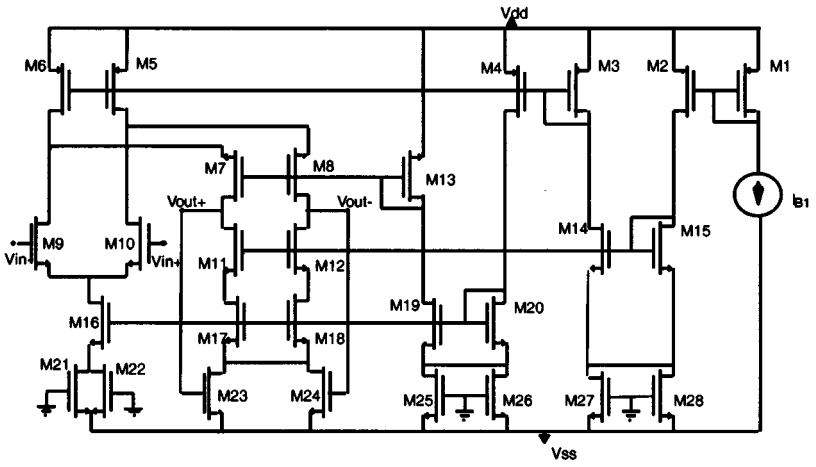

Fig. 12. Example circuit for analog circuit synthesis.

Table 3
Example of Analog Circuit Synthesis Results with FRIDGE and MAELSTROM

performance	specification	FRIDGE	MAELSTROM
nr of variables	-	10	31
CPU time	-	45 min	184 min
DC gain	≥ 70 dB	79 dB	76 dB
UGF	≥ 30 MHz	35 MHz	31 MHz
phase margin	≥ 60°	66°	61°
output swing	≥ 3.0 V	3.2 V	2.9 V
settling time	minimal	90 ns	62 ns

In the quest for industry-grade quality, most recent approaches, therefore, use complete SPICE simulations for all characteristics. To cut down on the large synthesis time, more efficient optimization algorithms are used and/or the simulations are executed as much as possible in parallel on a pool of workstations. In [93], the generalized boundary curve is used to determine the step length within an iterative trust-region optimization algorithm. Using the full nonlinear cost function based on the linearized objectives significantly reduces the total number of iterations in the optimization. The ANACONDA tool, on the other hand, uses a global optimization algorithm based on stochastic pattern search that inherently contains parallelism and, therefore, can easily be distributed over a pool of workstations, to try out and simulate 50 000 to 100 000 circuit candidates in a few hours [94]. MAELSTROM is the framework that provides the simulator encapsulation and the environment to distribute both the search tasks of the optimization algorithm as well as the circuit evaluations at every iteration of the optimization over parallel workstations in a network [95]. It uses another parallel global algorithm, a combined annealing-genetic algorithm, to produce fairly good designs in a few hours. These brute-force approaches require very little advance modeling to prepare for any new circuit topology and have the same accuracy as SPICE. Fig. 12 shows an example of an opamp circuit that has been synthesized with FRIDGE and MAELSTROM. The results are summarized in Table 3. In [96], ANACONDA/MAELSTROM, in combination with macromodeling techniques to bridge the hierarchical levels, was applied to an industrial-scale analog system (the equalizer/filter front end for an ADSL CODEC). The experiments demonstrated that the synthesis results are comparable to, or sometimes better than, manual design. Although appealing, these methods still have to be used with care by designers because the run times (and, therefore, also the debug time) remain long, and because the optimizer may easily produce improper designs if the right design constraints are not added to the optimization problem. Reducing the CPU time remains a challenging area for further research.

Other simulation-based approaches can be found in tools such as OAC [97], which is a specific nonlinear optimization tool for operational amplifiers and which is based on redesign starting from a previous design solution stored in the system's database. It also performs physical floorplanning during the optimization, which accounts for physical layout effects during circuit synthesis. A recent application of the simulation-based optimization approach to the high-level optimization of analog RF receiver front ends was presented in [98]. A dedicated RF front-end simulator was developed and used to calculate the ratio of the wanted signal to the total power of all unwanted signals (noise, distortion, aliasing, phase noise, etc.) in the frequency band of interest. An opti-

mization loop then determines the optimal specifications for the receiver subblocks such that the desired signal quality for the given application is obtained at the lowest possible power consumption for the overall front-end topology. Behavioral models and power estimators are used to evaluate the different front-end subblocks at this high architectural level.

In summary, the initial design systems like IDAC were too closed and restricted to their initial capabilities and, therefore, failed in the marketplace. The current trend is toward open analog design systems that allow the designer to easily extend and/or modify the design capabilities of the system without too much software overhead. Compared to the initial knowledge-based approaches, the big advantages of the more recent optimization-based approaches are their high flexibility and extendibility, both in terms of design objectives (by altering the cost function) and in terms of the ease to add new circuit schematics. Although some additional research is still needed, especially to reduce the CPU times, it can be concluded that a lot of progress has been achieved in the field of analog circuit synthesis during the past ten years. This has resulted in the development of several experimental analog synthesis systems, with which several designs have successfully been synthesized, fabricated, and measured. This has been accomplished not only for opamps, but also for filters [99] and data converters [100]. Based on these recent methods, several commercial tools are currently being developed that will be introduced in the marketplace in the near future.

Finally, it has to be added that industrial design practice not only calls for fully optimized nominal design solutions, but also expects high robustness and yield in the light of varying operating conditions (supply voltage or temperature variations) and statistical process tolerances and mismatches [101]. Techniques to analyze the impact of this on the yield or the capability index Cpk of the circuit [102] after the nominal design has been completed will be discussed in detail in Section III-E. Here, we briefly describe the efforts to integrate these considerations in the synthesis process itself. Yield and robustness precautions were already hardcoded in the design plans of IDAC [70], but are more difficult to incorporate in optimization-based approaches. Nevertheless, first attempts in this direction have already been presented. The ASTRX/OBLX tool has been extended with manufacturability considerations and uses a nonlinear infinite programming formulation to search for the worst case "corners" at which the evolving circuit should be evaluated for correct performance [103]. The approach has been successful in several test cases but does increase the required CPU time even further (roughly by $4\times$–$10\times$). Also, the OPTIMAN program has been extended by fully exploiting the availability of the analytic design equations to generate closed-form expressions for the sensitivities of the performances to the process parameters [104]. The impact of tolerances and mismatches on yield or Cpk can then easily be calculated at each optimization iteration, which then allows to synthesize the circuits simultaneously for performance and for manufacturability (yield or Cpk). The accuracy of the statistical predictions still has to be improved. The approach in [93] uses parameter distances as robustness objectives to obtain a nominal design that satisfies all specifications with as much safety margin as possible for process variations. The resulting formulation is the same as for design centering and can be solved efficiently using the generalized boundary curve. Design centering, however, still remains a second step after the nominal design. More research in this direction, therefore, is still needed.

D. Analog and Mixed-Signal Layout Synthesis

The next important step in the analog design flow of Fig. 4 after the circuit synthesis is the generation of the layout. The field of analog layout synthesis is more mature than circuit synthesis, in large part because it has been able to leverage ideas from the mature field of digital layout. Yet, real commercial solutions are only beginning to appear in recent years. Below we distinguish analog circuit-level layout synthesis, which has to transform a sized transistor-level schematic into a mask layout, and system-level layout assembly, in which the basic functional blocks are already laid out and the goal is to floorplan, place, and route them, as well as to distribute the power and ground connections.

1) Analog Circuit-Level Layout Synthesis: The earliest approaches to analog cell layout synthesis relied on **procedural module generation**, like in [105], in which the layout of the entire circuit was precoded in a software tool that generates the complete layout at run time for the actual parameter values entered. This approach is today frequently used during interactive manual layout for the single-keystroke generation of the entire layout of a single device or a special group of (e.g., matched) devices by means of parameterized procedural device generators. For circuits, however, the approach is not flexible enough, and large changes in the circuit parameters (e.g., device sizes) may result in inefficient area usage. In addition, a module generator has to be written and maintained for each individual circuit.

A related set of methods are called **template driven**. For each circuit, a geometric template (e.g., a sample layout [71] or a slicing tree [63]) is stored that fixes the relative position and interconnection of the devices. The layout is then completed by correctly generating the devices and the wires for the actual values of the design according to this fixed geometric template, thereby trying to use the area as efficiently as possible. These approaches work best when the changes in circuit parameters result in little need for global alterations in the general circuit layout structure, which is the case for instance during technology migration or porting of existing layouts, but which is not the case in general.

In practice, changes in the circuit's device sizes often require large changes in the layout structure in order to get the best performance and the best area occupation. The performance of an analog circuit is indeed impacted by the layout. Parasitics introduced by the layout, such as the parasitic wire capacitance and resistance or the crosstalk capacitance between two neighboring or crossing wires, can have a negative impact on the performance of analog circuits. It is, therefore, of utmost importance to generate analog

circuit layouts such that: 1) the resulting circuit still satisfies all performance specifications; and 2) the resulting layout is as compact as possible. This requires full-custom layout generation, which can be handled with a **macrocell-style layout strategy**. The terminology is borrowed from digital floorplanning algorithms, which manipulate flexible layout blocks (called "macros"), arrange them topologically and then route them. For analog circuits, the entities to be handled are structural groups of one single or a special grouping of devices (e.g., a matching pair of transistors). These "device-level macros" are to be folded, oriented, placed, and interconnected to make up a good overall layout. Note that many analog devices and special device groupings, even for the same set of parameters, can be generated in different geometrical variants, e.g., two matching devices can be laid out in interdigitated form, or stacked, or in a quad-symmetric fashion, etc. For each variant of each macrocell structure used, procedural module generators have to be developed to generate the actual layouts of the cells for a given set of parameter values. A drawback is that these generators have to be maintained and updated whenever the technology process changes, which creates some pressure to limit the number of different generators. Whatever the macrocells considered in a custom analog circuit layout synthesis tool, a placement routine optimally arranges the cells, while also selecting the most appropriate geometrical variant for each; a router interconnects them, and sometimes a compactor compacts the resulting layout, all while taking care of the many constraints like symmetry and matching typical for analog circuits, and also attending to the numerous parasitics and couplings to which analog circuits (unfortunately) are sensitive. This general analog circuit layout synthesis flow is shown in Fig. 13.

The need to custom optimize analog layouts led to the **optimization-based macrocell-place-and-route layout generation approaches** where the layout solution is not predefined by some template, but determined by an optimization program according to some cost function. This cost function typically contains minimum area and net length and adherence to a given aspect ratio, but also other terms could be added, and the user normally can control the weighting coefficients of the different contributions. The advantage of the optimization-based approaches is that they are generally applicable and not specific to a certain circuit, and that they are flexible in terms of performance and area as they find the most optimum solution at run time. The penalty to pay is their larger CPU times, and the dependence of the layout quality on the set-up of the cost function. Examples of such tools are ILAC [106] and the different versions of KOAN/ANAGRAM [107]. ILAC borrowed heavily from the best ideas from digital layout: efficient slicing-tree floorplanning with flexible blocks, global routing via maze routing, detailed routing via channel routing, area optimization via compaction [106]. The problem with the approach was that it was difficult to extend these primarily-digital algorithms to handle all the low-level geometric optimizations that characterize expert manual design. Instead, ILAC relied on a large, very sophisticated library of device generators.

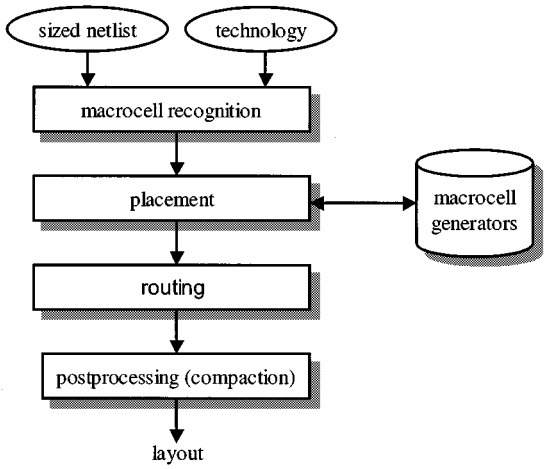

Fig. 13. General flow of an analog circuit layout synthesis tool.

ANAGRAM and its successor KOAN/ANAGRAM II kept the macrocell style, but reinvented the necessary algorithms from the bottom up, incorporating many manual design optimizations [107]–[109]. The device placer KOAN relied on a very small library of device generators and migrated important layout optimizations into the placer itself. KOAN, which was based on an efficient simulated annealing algorithm, could dynamically fold, merge, and abut MOS devices and, thus, discover desirable optimizations to minimize parasitic capacitance on the fly during optimization. Its companion, ANAGRAM II, was a maze-style detailed area router capable of supporting several forms of symmetric differential routing, mechanisms for tagging compatible and incompatible classes of wires (e.g., noisy and sensitive wires), parasitic crosstalk avoidance and over-the-device routing. Also, other device placers and routers operating in the macrocell-style have appeared (e.g., LADIES [110] and ALSYN [111]). Results from these tools can be quite impressive. For example, Fig. 14 shows two versions of the layout of an industrial 0.25-μm CMOS comparator. On the left is a manually created layout, on the right is a layout generated automically with a commercial tool operating in the macrocell style. The automatic layout compares well to the manual one.

An important improvement in the next generation of optimization-based layout tools was the shift from a rather qualitative consideration of analog constraints to an explicit quantitative optimization of the performance goals, resulting in the **performance-driven or constraint-driven approaches**. For example, KOAN maximized MOS drain-source merging during layout and ANAGRAM II minimized crosstalk, but without any specific, quantitative performance targets. The performance-driven approaches, on the other hand, explicitly quantify the degradation of the performance due to layout parasitics and the layout tools are driven such that this extra layout-induced performance degradation is within the margins allowed by the designer's performance specifications [112]. In this way, more optimum solutions can be found as the importance of each layout parasitic is weighed according to its impact on the circuit performance, and the tools can much better guarantee by construction that the circuit

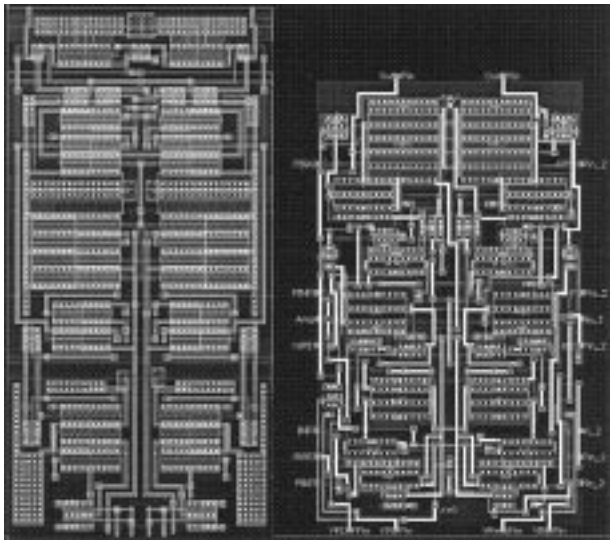

Fig. 14. Manual (left) versus automatic (right) layout for an industrial 0.25 μm CMOS analog comparator.

will meet the performance specifications also after the layout phase (if possible).

Tools that adopt this approach include the area router ROAD [113], the placement tool PUPPY-A [114] and the compaction tool SPARCS-A [115]. The routers ROAD [113] and ANAGRAM III [116] have a cost function that drives them such that they minimize the deviation from acceptable bounds on wire parasitics. These bounds are provided by designers or derived from the margins on the performance specifications via sensitivities. The LAYLA system [81], [117], [118] consists of performance-driven analog placement and routing tools that penalize excess layout-induced performance degradation by adding the excess degradation directly as an extra term to the cost function. Effects considered include for instance the impact of device merging, device mismatches, parasitic capacitance and resistance of each wire, parasitic coupling due to specific proximities, thermal gradients, etc. The router can manage not just parasitic wire sensititivies, but also yield and testability concerns. A layout generated by means of LAYLA was shown in Fig. 11.

In all the above tools, sensitivity analysis is used to quantify the impact on final circuit performance of low-level layout decisions and has emerged as the critical glue that links the various approaches being taken for circuit-level layout and for system assembly. An influential early formulation of the sensitivity analysis problem was [119], which not only quantified layout impacts on circuit performance, but also showed how to use nonlinear programming techniques to map these sensitivities into maximum bounds on parasitics, which serve as constraints for various portions of the layout task. Later approaches [117], however, showed that this intermediate mapping step may not be needed. Other work [120] showed how to extract critical constraints on symmetry and matching directly from a device schematic.

A recent innovation in CMOS analog circuit layout generation tools is the idea of separating the device placement into two distinct tasks: device stacking followed by stack placement. By rendering the circuit as an appropriate graph of connected drains and sources, it is possible to identify natural clusters of MOS devices that ought to be merged, called stacks, to minimize parasitic capacitance, instead of discovering these randomly over the different iterations of the placement optimization. The work in [121] presented an exact algorithm to extract all the optimal stacks, and dynamically choose the right stacking and the right placement of each stack throughout the placement optimization. Since the underlying algorithm has exponential time complexity, enumerating all stacks can be very time consuming. The work in [122] offered a variant that extracts one optimal set of stacks very fast. The idea is to use this in the inner loop of a placer to evaluate fast trial merges on sets of nearby devices.

One final problem in the macrocell place-then-route style is the separation of the placement and routing steps. The problem is to estimate how much wiring space to leave around each device for the subsequent routing. Too large estimates result in open space, too small estimates may block the router and require changes to the placement. One solution is to get better estimates by carrying out simultaneous placement and global routing, which has been implemented for slicing-style structures in [123]. An alternative is to use dynamic wire space estimation where space is created during routing when needed. Another strategy is analog compaction, where extra space is left during placement, which after routing is then removed by compaction. Analog compactors that maintain the analog constraints introduced by the previous layout steps were for instance presented in [115] and [124]. A more radical alternative is to perform simultaneous device place and route. An experimental version of KOAN [125] supported this by iteratively perturbing both the wires and the devices, but the method still has to be improved for large practical circuits.

Performance-driven macrocell-style custom analog circuit-level layout schemes are maturing nowadays, and the first commercial versions already started to be offered. Of course, there are still problems to solve. The wire space problem is one. Another open problem is "closing the loop" from circuit synthesis to circuit layout, so layouts that do not meet the specifications can, if necessary, cause actual circuit design changes (via circuit resynthesis). Even if a performance-driven approach is used, which should generate layouts correct by construction, circuit synthesis needs accurate estimates of circuit wiring loads to obtain good sizing results, and circuit synthesis needs to leave sufficient performance margins for the layout-induced performance degradation later on. How to control this loop, and how to reflect layout concerns in synthesis and synthesis concerns in layout remain difficult.

2) Mixed-Signal System Layout Assembly: A mixed-signal system is a set of analog and digital functional blocks. System-level layout assembly means floorplanning, placement, and global and detailed routing (including the power grid) of the entire system, where the layouts of the individual blocks are generated by lower-level tools such as discussed in the previous section for the analog circuits. In

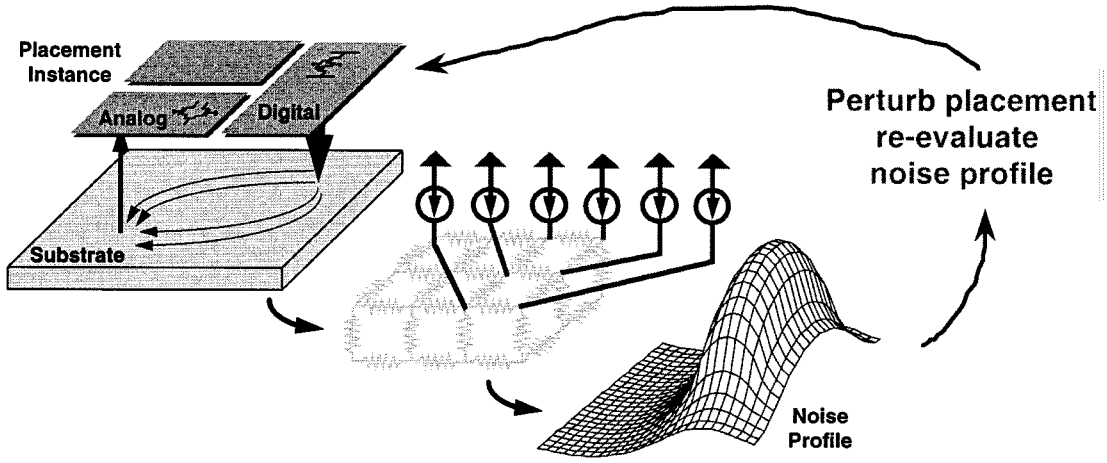

Fig. 15. Principal flow of the WRIGHT mixed-signal floorplanner that incorporates a fast substrate noise coupling evaluator.

addition to sensitivities to wire parasitics, an important new problem in mixed-signal systems is the coupling between digital switching noise and sensitive analog circuits (for instance, capacitive crosstalk or substrate noise couplings).

As at the circuit level, procedural layout generation remains a viable alternative for well-understood designs with substantial regularity (e.g., switched-capacitor filters [126]). More generally though, work has focused on custom placement and routing at the block level. For row-based layout, an early elegant solution to the coupling problem was the segregated-channels idea of [127] to alternate noisy digital and sensitive analog wiring channels in a row-based cell layout. The strategy constrains digital and analog signals never to be in the same channel, and remains a practical solution when the size of the layout is not too large. For large designs, analog channel routers were developed. In [128], it was observed that a well-known digital channel routing algorithm, based on a gridless constraint-graph formulation, could easily be extended to handle critical analog problems that involve varying wire widths and wire separations needed to isolate noisy and sensitive signals. The channel router ART extended this strategy to handle complex analog symmetries in the channel, and the insertion of shields between incompatible signals [129].

The WREN [130] and WRIGHT [131] tools generalized these ideas to the case of arbitrary layouts of mixed functional blocks. WREN comprises both a mixed-signal global router and channel router [130]. The tool introduced the notion of signal-to-noise ratio (SNR)-style constraints for incompatible signals, and both the global and detailed routers strive to comply with designer-specified noise rejection limits on critical signals. WREN incorporates a constraint mapper that transforms input noise rejection constraints from the across-the-whole-chip form used by the global router into the per-channel per-segment form necessary for the channel router. WRIGHT on the other hand uses simulated annealing to floorplan the blocks, but with an integrated fast substrate-noise-coupling evaluator so that a simplified view of substrate noise influences the floorplan [131]. Fig. 15 shows the flow of this tool. Accurate methods to analyze substrate couplings have been presented in Section III-A. In the frame of layout synthesis tools, however, the CPU times of these techniques are prohibitive and there is a need for fast yet accurate substrate-noise-coupling evaluators to explore alternative layout solutions.

Another important task in mixed-signal system layout is power grid design. Digital power grid layout schemes usually focus on connectivity, pad-to-pin ohmic drop, and electromigration effects. But these are only a small subset of the problems in high-performance mixed-signal chips, which feature fast-switching digital systems next to sensitive analog parts. The need to mitigate unwanted substrate interactions, the need to handle arbitrary (nontree) grid topologies, and the need to design for transient effects such as current spikes are serious problems in mixed-signal power grids. The RAIL system [132], [133] addresses these concerns by casting mixed-signal power grid synthesis as a routing problem that uses fast asymptotic-waveform-evaluation-based [38] linear system evaluation to electrically model the entire power grid, package and substrate in the inner loop of grid optimization. Fig. 16 shows an example RAIL redesign of a data channel chip in which a demanding set of dc, ac, and transient performance constraints were met automatically.

Most of these mixed-signal system-level layout tools are of recent vintage, but because they often rely on mature core algorithms from similar digital layout problems, many have been prototyped both successfully and quickly. Although there is still much work to be done to enhance existing constraint mapping strategies and constraint-based layout tools to handle the full range of industrial concerns, the progress obtained opens possibilities for commercialization activities in the near future to make these tools available to all practicing mixed-signal designers.

E. Yield Estimation and Optimization

The manufacturing of integrated circuits suffers from statistical fluctuations inherent to the fabrication process itself [101]. These fluctuations can be local (e.g., lithographic spots) or global (e.g., gate oxide gradients). These

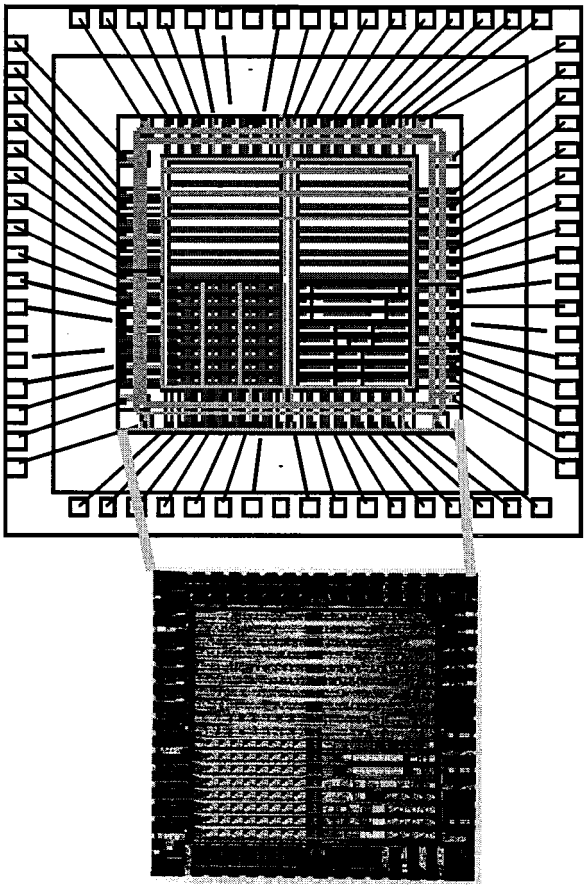

Fig. 16. RAIL power grid design for IBM data channel chip.

process-induced deformations result in deviations of the actual performances of the fabricated ICs from the expected performances. Some of these deformations result in the fabricated IC not having the expected functionality. This is then called a *structural failure*. Others cause the fabricated IC to have, despite the correct functionality, observed performances far from the targeted values. This is then called a *hard performance failure*. Both failures are also termed *catastrophic failures*, caused by a catastrophic fault. Other process deformations result in the observed performances deviating only slightly around the targeted values due to the statistical tolerances (interdie variations) and mismatches (intradie variations) of the device parameters. This is then called a *parametric* (*or soft*) *performance failure*, caused by a parametric fault.

Both catastrophic and parametric faults cause a fraction of the fabricated ICs to have performances that do not meet the required specifications. The ratio of accepted to the total number of fabricated ICs is called the **yield**. If the yield is significantly less than 100%, this implies a financial loss to the IC manufacturer. Therefore it is important to already calculate and maximize the manufacturing yield during the design stage. This is called **design for manufacturability**, which implies techniques for yield estimation and yield optimization. Even more expensive are the field failures that show up when the IC is in use in a product in the field, for instance when the IC is used under extreme operating conditions like high temperatures. To try to avoid this, the design has to be made as robust as possible. This is called **design for robustness or design for quality**, which implies techniques for variability minimization and design centering. Both aspects are of course interrelated and can be captured in a characteristic like the capability index **Cpk** [102].

Due to the fluctuations of the device parameters, the performance characteristics of the circuit will show fluctuations. The corresponding parametric yield is the integral over the acceptability region (i.e., the region where all specifications are satisfied) of the joint probability density function of the fluctuating parameters. This yield can be calculated in both the device parameter space or the circuit performance space. Its calculation, however, is complicated by the fact that in either space one of the two elements is not known explicitly: the statistical fluctuations are known in the device parameter space but not in the circuit performance space, whereas the acceptability region is known in the performance space but not in the parameter space [134]. The relation between the device parameter space and the circuit performances depends on the nominal design point chosen for the circuit, but is in general a nonlinear transformation that is not known explicitly. This relation can for instance be derived on a point by point basis using SPICE simulations. All this makes yield estimation a difficult task where the different techniques trade off accuracy versus CPU time in a different way.

A simple approach often used in practice is **worst case analysis** [102]. Instead of doing a true statistical analysis, some worst case combinations (e.g., slow p-type devices, fast n-type devices, etc.) of the device parameters are used to calculate the "worst case" corners of the performances. Problems are that this approach can lead to large overestimations and that for analog circuits it is usually not known *a priori* which combinations of the device parameters result in the worst case corners. A deterministic technique based on sequential quadratic programming (SQP) to calculate the worst case device model parameters and worst case operating conditions for each performance separately has been described in [135]. The corresponding worst case distances can be used to measure the yield and the robustness of the design.

The most straightforward statistical approach is to use **Monte-Carlo simulations**, in which a large number of samples of the device parameters are generated according to the given statistics. The samples must include the correlations, but principal component analysis (PCA) techniques can be used to generate uncorrelated parameter sets. For each generated sample set a SPICE simulation is performed and all resulting performance data are combined to derive the statistics of the circuit performances. Since this process requires many circuit simulations, especially when the confidence factor on the yield or Cpk estimate has to be small, this Monte-Carlo approach is very time-consuming.

The CPU time can be reduced with the response surface method [136], which works in two steps. First, the parameter space is sampled with controlled simulations according to some design-of-experiments (DOE) scheme. For each performance characteristic a response surface is then constructed by fitting a simple function (e.g., a first- or

second-order polynomial) of the device parameters to the simulated performance data. By initial screening, unimportant device parameters can be eliminated from each model. Then, in the second step, the evaluation of these simple response surface models replaces full circuit simulations during the yield or Cpk calculation, for instance using the Monte-Carlo method. The only limitation of this approach is the accuracy of the response surface models. Nevertheless, various forms of Monte-Carlo simulation—with or without response surface acceleration technique—remain staples for almost all analog design methodologies. A rigorous formulation to estimate the parametric yield of a circuit affected by mismatches has been presented in [137]. The approach, however, does not include yield due to defects.

Finally, if the calculated yield or Cpk is not sufficient, then the design will have to be altered in order to increase the yield or the design robustness [102], [134]. By changing the nominal design point, either the probability density function in the performance space, or the acceptability region in the device parameter space will be modified, and hence the yield/Cpk. The goal is then to modify the nominal design point such that the yield is maximized, or the variability of the circuit performance is minimized, or the design is centered. The most direct approach is to put one of the above yield/Cpk estimation routines in an optimization loop and directly try to maximize the yield/Cpk. Needless to say, this is extremely time consuming. Here again, the response surface technique can be used to reduce the number of required simulations, by using the technique in two subsequent phases and generating an explicit response surface model for the yield or Cpk as a function of the nominal design point parameters. An alternative is the technique of design centering where the design is centered as much as possible within the acceptability region. Geometric approximation techniques such as simplicial approximation, etc., have to be used to approximate the acceptability region in the device parameter space, but these techniques suffer from a bad computational complexity. The approach in [93] uses the worst case distances as robustness objectives during design centering, and applies the generalized boundary curve technique to reduce the number of simulations required. The challenge is also to incorporate yield and robustness optimization as an integrated part of circuit synthesis, instead of considering it as an separate step to be performed after a first nominal-only circuit sizing. First attempts in this direction have already been presented in Section III-C [103], [104], but the execution times still have to be reduced further.

F. Analog and Mixed-Signal Testing and Design for Testability

The final step in the IC production cycle is the testing of the IC [138], [139]. The objective of testing is to verify whether the fabricated IC meets the specifications it was designed for or not. The cost of testing is a considerable part of the final cost of an IC, and the later in the production cycle a certain defect is detected, the more expensive the financial loss. The effective cost of a product is therefore related to the quality of the applied tests and the time necessary to generate and apply the tests. In mixed-signal designs, an additional complication is that, although analog circuits constitute only a small fraction of the die area, they are responsible for the largest fraction of the overall test cost. This is not only because they require more expensive mixed- signal test equipment, but also because they require longer test times than the digital circuitry and because there are no structured test waveform generation tools nor structured design for test methodologies.

The testing of an IC is complicated by the limited pin count, which means that only a limited number of nodes can be accessed from the outside. In order to increase the controllability and observability of internal nodes, design for testability (DfT) measures have to be included in the design and layout of the IC. A second problem critical in analog designs is the presence of statistical process parameter fluctuations, which make multiple fabricated samples of the same circuit showing a statistical spread of responses around the nominal response (called a *tolerance box*). Due to this spreading, the tolerance boxes of the fault-free and certain faulty circuits may overlap for the given test set, creating so called ambiguity regions where the given test cannot uniquely distinguish between fault-free and faulty devices. The test may therefore result in some percentage of undecisive or wrong test decisions, which of course has to be reduced as much as possible. Fortunately, the situation is changing and the field of analog and mixed-signal testing is characterized by many interesting developments, which will only be touched upon here briefly. The reader is referred to [138]–[140] for more details.

For **fault detection** in go/no-go production testing, test cost and test time are critical. Two basic approaches can be distinguished. The traditional test approach is **specification based**, also called functional testing. Tests are applied to check the functionality of the circuit under test against the specifications. For instance, for an analog-to-digital converter, these are the traditional tests such as the histogram test to derive the integral and differential nonlinearity (INL and DNL) and the effective number of bits (ENOB). The problems with this approach are that there is no quantification of how necessary and how sufficient these tests are for detecting whether a circuit is fault-free or not, and that this test generation process is difficult to automate. A more recent approach, therefore, is **defect-based** analog testing, mimicking the approach in use for a long time in digital testing. In this case, tests are applied to detect whether specific faults are present in the fabricated circuit structure or not. This requires that the faults that are being tested for are collected in advance in a so-called fault dictionary, which implies the use of a certain fault model and of fault simulation. Unlike in the digital world, however, there is no common agreement in the analog design and test community on the fault models to be used. For instance, a spot defect shorting two neighboring wires can be represented as an ideal short of the two corresponding nodes, or as a resistor between the two nodes with a value of 0.1 Ω, or a resistor with a value of 1 Ω, etc. All this makes it hard to calculate a "fault coverage" characteristic to qualify test

sets. In addition, not all possible faults can occur in practice. For instance, assuming all possible shorts between any two nodes in the circuit and all possible opens for all interconnections in the circuit may lead to unrealistically large fault lists. A more efficient technique is inductive fault analysis [140] that generates a list of physically realistic faults to be tested for by sprinkling defects across the actual layout and extracting possible faults from this. In any case, whatever the fault list, each faulty circuit considered has to be simulated during the design stage to build up the fault dictionary, which is extremely time consuming. Therefore, techniques to speed up fault simulations have received large attention in recent years [141].

During **fault diagnosis** in the initial IC prototype characterization and validation phase the test problem is even more complicated: not only must it be detected whether there is a fault or not in the circuit, but also the location of the fault has to be identified to diagnose the error. This is more difficult as there may also be ambiguity regions between the responses of circuits with different faults, making it impossible to distinguish between the ambiguous cases for the given test set. Two basic approaches are possible [138]. In **simulation before test** (SBT), the fault-free and faulty circuits are simulated beforehand and the responses are stored in a fault dictionary. During testing the measured response is then compared to the signatures in the fault dictionary to identify the most likely case. The second approach is **simulation after test** (SAT), where the circuit is measured first and then the circuit parameters are reconstructed from the measurement results and compared to the fault-free values. It is not always possible, however, to uniquely solve the value of each element out of a given measurement set, making it necessary to use optimization techniques to estimate the most likely element values from the measurements.

Concerning the tests themselves, both dc, ac (frequency), or transient tests can be carried out, and the measured signals can be node voltages, pin currents (e.g., the power-supply current—the equivalent of digital IDDQ testing) or derived characteristics. The goal of **test waveform generation** is to determine which tests have to be carried out (which stimuli to be applied, which signals to be measured) in order to have maximum fault coverage at a minimum test cost. In general, this has to be formulated as an optimization problem and some interesting approaches have been presented in recent years [142], [143]. In order to overcome the problem of the limited accessibility in ICs, **design for testability** measures have to be taken to improve the controllability and observability by propagating test signals from the external IC pins to and from internal, otherwise inaccessible nodes [140]. These measures can for instance take the form of extra test pins or extra test structures [e.g., (de)multiplexers] to isolate and individually test the different analog blocks in the system [144], analog scan path techniques to read out selected test node signals into a serial scan path [145], or extra circuitry to reconfigure the circuit in the test mode, e.g., use a feedback structure to reconfigure the circuit into an oscillator mode where the oscillation frequency then indicates whether the circuit is faulty or fault-free [146].

With the move toward systems on a chip where the analog circuitry is deeply embedded into the system, the limited accessibility problem will become more and more stringent. An appealing alternative for such systems is the use of **built-in self test** (BIST). In this case, the generation of the test waveforms, the capturing of the test responses, and their comparison to the expected good responses are all performed on-chip, resulting in the circuit autonomously returning a pass or fail signal, at the expense of some chip area overhead. The response comparison has to consider tolerances both on the stored good signature and on the measured signals in order to avoid false rejections and escapes. Although most mixed-signal BIST schemes today have a restricted applicability, some noteworthy approaches have already been presented, most of them exploiting the presence of on-chip data converters and surrounding digital circuitry. In the MADBIST scheme, after self-testing the digital circuitry, the D/A converter is configured in oscillator mode to test the A/D converter, after which the D/A converter and then the other analog circuitry is tested [147]. In the HBIST scheme for so-called "discretized analog" applications (i.e., applications consisting of the sequence: analog in–ADC–DSP–DAC–analog out), the analog output in test mode is fed back to the analog input, to create a loop that starts and ends at the central DSP part [148]. After first self-testing the DSP part, the analog loop can then be tested while test signals are provided and processed by the DSP part. With the move toward systems on a chip and the paradigm of building systems by mixing and matching IP blocks, the role of analog BIST schemes will become more and more important in the future, and more research in this direction is to be expected. Also, the IEEE P1149.4 standardized mixed-signal test bus may create opportunities here [149].

IV. Conclusion

The increasing levels of integration in the microelectronics industry, now approaching the era of systems on a chip, has also brought about the need for systematic design methodologies and supporting CAD tools that increase the productivity and improve the quality in the design of analog and mixed-signal integrated circuits and systems. This survey paper has presented an overview of the current context and state of the art in the field of analog and mixed-signal CAD tools. After introducing the industrial context and the design flow, an overview has been given of the progress in analog and mixed-signal simulation and modeling, symbolic analysis, analog circuit synthesis and optimization, analog and mixed-signal layout generation, yield analysis and design centering, as well as analog and mixed-signal test.

Most progress has been obtained in the field of analog, mixed analog–digital and multilevel analog simulation, where many commercial solutions are available. Also, the standardization of analog and mixed-signal hardware description languages is approaching completion. In the field of custom analog circuit and layout synthesis, substantial progress at the research level has been achieved over the

past decade, despite the dearth of commercial offerings yet. Cast mostly in the form of numerical and combinatorial optimization tasks, linked by various forms of sensitivity analysis and constraint mapping, leveraged by ever faster workstations, some of these tools show glimmers of practical application, and commercial startups have embarked to bring these to industrial practice. Also, in the field of analog and mixed-signal test new ideas have been developed, but they are striving for industrial acceptance now.

One conclusion is clear: In order to meet the economic constraints (time to market, cost, quality) of future semiconductor products, analog design will have to be carried out in a much more systematic and structured way, supported by methodologies and tools that fit in the overall system design flow. Although research in academia has not fully solved all the relevant problems yet, this paper has shown that real progress has been made over the last decade. Therefore, given the current market pressures and given the existence of core design technology in academia, we can hope that commercial offerings will soon follow and that analog designers will finally get the boost in productivity needed to take them from the dark ages of analog black magic to the bright future of integrated systems on a chip. In the emerging era of combined information and communication technologies and ubiquitous computing, we need good tools for analog circuits more than ever.

REFERENCES

[1] "Systems-on-a-chip," in *IEEE Int. Solid-State Circuits Conf. (ISSCC)*, 1996.

[2] J. Liang, "Mixed-signal IC market to surpass $10 billion in 1997 and $22 billion by 2001," Dataquest Rep., Jan. 1998.

[3] *The National Technology Roadmap for Semiconductors*: Semiconductor Industry Association (SIA), 1994.

[4] *Virtual Socket Interface Architecture Document*: Version 1.0, VSI Alliance, 1997.

[5] H. Chang *et al.*, *Surviving the SOC Revolution—A Guide to Platform-Based Design*. Norwell, MA: Kluwer, 1999.

[6] L. Nagel, "SPICE2: A computer program to simulate semiconductor circuits," Electronics Research Lab., Univ. Calif., Berkeley, Memo UCB/ERL M520, May 1975.

[7] L. R. Carley, G. Gielen, R. Rutenbar, and W. Sansen, "Synthesis tools for mixed-signal ICs: Progress on frontend and backend strategies," in *Proc. ACM/IEEE Design Automation Conf. (DAC)*, 1996, pp. 298–303.

[8] *IEEE Standard VHDL 1076.1 Language Reference Manual—Analog and Mixed-Signal Extensions to VHDL 1076*: IEEE 1076.1 Working Group, July 1997.

[9] *Verilog-A: Language Reference Manual: Analog Extensions to Verilog HDL*: Version 0.1, Open Verilog International, Jan. 1996.

[10] R. Harjani, R. Rutenbar, and L. R. Carley, "OASYS: A framework for analog circuit synthesis," *IEEE Trans. Computer-Aided Design*, vol. 8, pp. 1247–1265, Dec. 1989.

[11] G. Gielen, K. Swings, and W. Sansen, "Open analog synthesis system based on declarative models," in *Analog Circuit Design*, J. Huijsing, R. van der Plassche, and W. Sansen, Eds. Norwell, MA: Kluwer, 1993, pp. 421–445.

[12] S. Donnay *et al.*, "Using top–down CAD tools for mixed analog/digital ASICs: A practical design case," *Kluwer Int. J. Analog Integrated Circuits Signal Processing (Special Issue on Modeling and Simulation of Mixed Analog–Digital Systems)*, vol. 10, pp. 101–117, June–July 1996.

[13] H. Chang *et al.*, "A top–down, constraint-driven design methodology for analog integrated circuits," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 1992, pp. 8.4.1–8.4.6.

[14] E. Malavasi *et al.*, "A top–down, constraint-driven design methodology for analog integrated circuits," in *Analog Circuit Design*, J. Huijsing, R. van der Plassche, and W. Sansen, Eds. Norwell, MA: Kluwer, 1993, ch. 13, pp. 285–324.

[15] H. Chang *et al.*, *A Top–Down, Constraint-Driven Design Methodology for Analog Integrated Circuits*. Norwell, MA: Kluwer, 1997.

[16] L. Nagle and R. Rohrer, "Computer analysis of nonlinear circuits, excluding radiation (CANCER)," *IEEE J. Solid-State Circuits*, vol. SSC-6, pp. 166–182, Aug. 1971.

[17] A. Vladimirescu, *The SPICE Book*. New York: Wiley, 1994.

[18] D. Foty, *MOSFET Modeling with SPICE*. Englewood Cliffs, NJ: Prentice-Hall, 1997.

[19] C. Enz, "MOS transistor modeling for RF integrated circuit design," in *Proc. IEEE Custom Integrated Circuit Conf. (CICC)*, 2000, pp. 189–196.

[20] R. Saleh, B. Antao, and J. Singh, "Multi-level and mixed-domain simulation of analog circuits and systems," *IEEE Trans. Computer-Aided Design*, vol. 15, pp. 68–82, Jan. 1996.

[21] A. Vachoux, J.-M. Bergé, O. Levia, and J. Rouillard, Eds., *Analog and Mixed-Signal Hardware Description Languages*. Norwell, MA: Kluwer, 1997.

[22] G. Casinovi and A. Sangiovanni-Vincentelli, "A macromodeling algorithm for analog circuits," *IEEE Trans. Computer-Aided Design*, vol. 10, pp. 150–160, Feb. 1991.

[23] Y.-C. Ju, V. Rao, and R. Saleh, "Consistency checking and optimization of macromodels," *IEEE Trans. Computer-Aided Design*, pp. 957–967, Aug. 1991.

[24] B. Antao and F. El-Turky, "Automatic analog model generation for behavioral simulation," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, May 1992, pp. 12.2.1–12.2.4.

[25] C. Borchers, L. Hedrich, and E. Barke, "Equation-based behavioral model generation for nonlinear analog circuits," in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, 1996, pp. 236–239.

[26] S. Fang, Y. Tsividis, and O. Wing, "SWITCAP: A switched-capacitor network analysis program—Part I: Basic features," *IEEE Circuits Syst. Mag.*, vol. 5, pp. 4–10, Sept. 1983.

[27] S. Fang, Y. Tsividis, and O. Wing, "SWITCAP: A switched-capacitor network analysis program—Part I: Advanced applications," *IEEE Circuits Syst. Mag.*, vol. 5, pp. 41–46, Sept. 1983.

[28] J. Vandewalle, H. De Man, and J. Rabaey, "Time, frequency, and Z-domain modified nodal analysis of switched-capacitor networks," *IEEE Trans. Circuits Syst.*, vol. CAS-28, pp. 186–195, Mar. 1981.

[29] V. Dias, V. Liberali, and F. Maloberti, "TOSCA: A user-friendly behavioral simulator for oversampling A/D converters," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 1991, pp. 2677–2680.

[30] K. Kundert, "Simulation methods for RF integrated circuits," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1997, pp. 752–765.

[31] R. Telichevesky, K. Kundert, I. Elfadel, and J. White, "Fast simulation algorithms for RF circuits," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 1996, pp. 437–444.

[32] K. Kundert, "Introduction to RF simulation and its applications," *IEEE J. Solid-State Circuits*, vol. 34, pp. 1298–1319, Sept. 1999.

[33] P. Feldmann and J. Roychowdhury, "Computation of circuit waveform envelopes using an efficient, matrix-decomposed harmonic balance algorithm," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1996, pp. 295–300.

[34] J. Crols, S. Donnay, M. Steyaert, and G. Gielen, "A high-level design and optimization tool for analog RF receiver front-ends," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1995, pp. 550–553.

[35] A. Demir and A. Sangiovanni-Vincentelli, *Analysis and Simulation of Noise in Nonlinear Integrated Circuits and Systems*. Norwell, MA: Kluwer, 1998.

[36] J. Phillips and K. Kundert, "Noise in mixers, oscillators, samplers, and logic: An introduction to cyclostationary noise," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 2000, pp. 431–439.

[37] L. Pileggi, "Coping with RC(L) interconnect design headaches," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1995, pp. 246–253.

[38] L. Pillage and R. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 352–366, Apr. 1990.

[39] L. Silveira *et al.*, "A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of RLC circuits," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1996, pp. 288–294.

[40] M. Kamon, S. McCormick, and K. Shepard, "Interconnect parasitic extraction in the digital IC design methodology," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1999, pp. 223–230.
[41] N. Verghese and D. Allstot, "Rapid simulation of substrate coupling effects in mixed-mode IC's," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 1993, pp. 18.3.1–18.3.4.
[42] R. Gharpurey and R. Meyer, "Modeling and analysis of substrate coupling in integrated circuits," *IEEE J. Solid-State Circuits*, vol. 31, pp. 344–353, Mar. 1996.
[43] N. Verghese, T. Schmerbeck, and D. Allstot, *Simulation Techniques and Solutions for Mixed-Signal Coupling in Integrated Circuits*. Norwell, MA: Kluwer, 1995.
[44] T. Blalack, "Design techniques to reduce substrate noise," in *Advances in Analog Circuit Design*, Huijsing, van de Plassche, and Sansen, Eds. Norwell, MA: Kluwer, 1999, pp. 193–217.
[45] J. Costa, M. Chou, and L. Silveira, "Efficient techniques for accurate modeling and simulation of substrate coupling in mixed-signal ICs," *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 597–607, May 1999.
[46] E. Charbon, R. Gharpurey, R. Meyer, and A. Sangiovanni-Vincentelli, "Substrate optimization based on semi-analytical techniques," *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 172–190, Feb. 1999.
[47] M. van Heijningen, M. Badaroglu, S. Donnay, M. Engels, and I. Bolsens, "High-level simulation of substrate noise generation including power supply noise coupling," in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, 2000, pp. 446–451.
[48] G. Gielen, P. Wambacq, and W. Sansen, "Symbolic analysis methods and applications for analog circuits: A tutorial overview," *Proc. IEEE*, vol. 82, pp. 287–304, Feb. 1994.
[49] F. Fernández, A. Rodríguez-Vázquez, J. Huertas, and G. Gielen, *Symbolic Analysis Techniques—Applications to Analog Design Automation*. Piscataway, NJ: IEEE Press, 1998.
[50] P. Wambacq, F. Fernández, G. Gielen, W. Sansen, and A. Rodríguez-Vázquez, "Efficient symbolic computation of approximated small-signal characteristics," *IEEE J. Solid-State Circuits*, vol. 30, pp. 327–330, Mar. 1995.
[51] G. Gielen, H. Walscharts, and W. Sansen, "ISAAC: A symbolic simulator for analog integrated circuits," *IEEE J. Solid-State Circuits*, vol. 24, pp. 1587–1596, Dec. 1989.
[52] P. Wambacq, G. Gielen, P. Kinget, and W. Sansen, "High-frequency distortion analysis of analog integrated circuits," *IEEE Trans. Circuits Syst. II*, vol. 46, pp. 335–345, Mar. 1999.
[53] S. Seda, M. Degrauwe, and W. Fichtner, "A symbolic analysis tool for analog circuit design automation," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1988, pp. 488–491.
[54] F. Fernández, A. Rodríguez-Vázquez, and J. Huertas, "Interactive ac modeling and characterization of analog circuits via symbolic analysis," *Kluwer Int. J. Analog Integrated Circuits Signal Processing*, vol. 1, pp. 183–208, Nov. 1991.
[55] Q. Yu and C. Sechen, "A unified approach to the approximate symbolic analysis of large analog integrated circuits," *IEEE Trans. Circuits Systems I*, vol. 43, pp. 656–669, Aug. 1996.
[56] W. Daems, G. Gielen, and W. Sansen, "Circuit complexity reduction for symbolic analysis of analog integrated circuits," in *Proc. IEEE/ACM Design Automation Conf.*, 1999, pp. 958–963.
[57] J. Hsu and C. Sechen, "DC small signal symbolic analysis of large analog integrated circuits," *IEEE Trans. Circuits Systems I*, vol. 41, pp. 817–828, Dec. 1994.
[58] J. Starzyk and A. Konczykowska, "Flowgraph analysis of large electronic networks," *IEEE Trans. Circuits Syst.*, vol. CAS-33, pp. 302–315, Mar. 1986.
[59] O. Guerra, E. Roca, F. Fernández, and A. Rodríguez-Vázquez, "A hierarchical approach for the symbolic analysis of large analog integrated circuits," in *Proc. IEEE Design Automation and Test in Europe Conf. (DATE)*, 2000, pp. 48–52.
[60] C.-J. Shi and X.-D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams," *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 1–18, Jan. 2000.
[61] X.-D. Tan and C.-J. Shi, "Hierarchical symbolic analysis of analog integrated circuits via determinant decision diagrams," *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 401–412, April 2000.
[62] F. El-Turky and E. Perry, "BLADES: An artificial intelligence approach to analog circuit design," *IEEE Trans. Computer-Aided Design*, vol. 8, pp. 680–691, June 1989.
[63] H. Koh, C. Séquin, and P. Gray, "OPASYN: A compiler for CMOS operational amplifiers," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 113–125, Feb. 1990.
[64] P. Veselinovic *et al.*, "A flexible topology selection program as part of an analog synthesis system," in *Proc. IEEE Eur. Design Test Conf. (ED&TC)*, 1995, pp. 119–123.
[65] R. Harjani and J. Shao, "Feasibility and performance region modeling of analog and digital circuits," *Kluwer Int. J. Analog Integrated Circuits Signal Processing*, vol. 10, pp. 23–43, Jan. 1996.
[66] P. Maulik, L. R. Carley, and R. Rutenbar, "Simultaneous topology selection and sizing of cell-level analog circuits," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 401–412, Apr. 1995.
[67] Z. Ning *et al.*, "SEAS: A simulated evolution approach for analog circuit synthesis," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 1991, pp. 5.2.1–5.2.4.
[68] W. Kruiskamp and D. Leenaerts, "DARWIN: CMOS opamp synthesis by means of a genetic algorithm," in *Proc. ACM/IEEE Design Automation Conf. (DAC)*, 1995, pp. 550–553.
[69] G. Gielen and W. Sansen, *Symbolic Analysis for Automated Design of Analog Integrated Circuits*. Norwell, MA: Kluwer, 1991.
[70] M. Degrauwe *et al.*, "IDAC: An interactive design tool for analog CMOS circuits," *IEEE J. Solid-State Circuits*, vol. 22, pp. 1106–1115, Dec. 1987.
[71] G. Beenker, J. Conway, G. Schrooten, and A. Slenter, "Analog CAD for consumer ICs," in *Analog Circuit Design*, J. Huijsing, R. van der Plassche, and W. Sansen, Eds. Norwell, MA: Kluwer, 1993, pp. 347–367.
[72] R. Henderson *et al.*, "A spreadsheet interface for analog design knowledge capture and reuse," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 1993, pp. 13.3.1–13.3.4.
[73] N. Horta, J. Franca, and C. Leme, "Automated high level synthesis of data conversion systems," in *Analogue–Digital ASICs—Circuit Techniques, Design Tools and Applications*, Soin, Maloberti, and Franca, Eds. Stevenage, U. K.: Peregrinus, 1991.
[74] J. Vital and J. Franca, "Synthesis of high-speed A/D converter architectures with flexible functional simulation capabilities," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 1992, pp. 2156–2159.
[75] C. Toumazou and C. Makris, "Analog IC design automation—I: Automated circuit generation: New concepts and methods," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 218–238, Feb. 1995.
[76] C. Makris and C. Toumazou, "Analog IC design automation—II: Automated circuit correction by qualitative reasoning," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 239–254, Feb. 1995.
[77] B. Sheu, A. Fung, and Y.-N. Lai, "A knowledge-based approach to analog IC design," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 256–258, Feb. 1988.
[78] J. Harvey, M. Elmasry, and B. Leung, "STAIC: An interactive framework for synthesizing CMOS and BiCMOS analog circuits," *IEEE Trans. Computer-Aided Design*, vol. 11, pp. 1402–1416, Nov. 1992.
[79] G. Gielen, H. Walscharts, and W. Sansen, "Analog circuit design optimization based on symbolic simulation and simulated annealing," *IEEE J. Solid-State Circuits*, vol. 25, pp. 707–713, June 1990.
[80] K. Swings and W. Sansen, "DONALD: A workbench or interactive design space exploration and sizing of analog circuits," in *Proc. IEEE Eur. Design Automation Conf. (EDAC)*, 1991, pp. 475–479.
[81] K. Lampaert, G. Gielen, and W. Sansen, *Analog Layout Generation for Performance and Manufacturability*. Norwell, MA: Kluwer, 1999.
[82] G. Gielen *et al.*, "An analog module generator for mixed analog/digital ASIC design," *Wiley Int. J. Circuit Theory Applicat.*, vol. 23, pp. 269–283, July–Aug. 1995.
[83] F. Medeiro, B. Pérez-Verdú, A. Rodríguez-Vázquez, and J. Huertas, "A vertically-integrated tool for automated design of $\Sigma\Delta$ modulators," *IEEE J. Solid-State Circuits*, vol. 30, pp. 762–772, July 1995.
[84] A. Doboli, A. Nunez-Aldana, N. Dhanwada, S. Ganesan, and R. Vemuri, "Behavioral synthesis of analog systems using two-layered design space exploration," in *Proc. ACM/IEEE Design Automation Conf. (DAC)*, 1999, pp. 951–957.
[85] F. Leyn, G. Gielen, and W. Sansen, "An efficient dc root solving algorithm with guaranteed convergence for analog integrated CMOS circuits," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1998, pp. 304–307.
[86] M. Hershenson, S. Boyd, and T. Lee, "GPCAD: A tool for CMOS op-amp synthesis," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1998, pp. 296–303.

[87] M. Hershenson, S. Mohan, S. Boyd, and T. Lee, "Optimization of inductor circuits via geometric programming," in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, 1999, pp. 994–998.
[88] S. Director and R. Rohrer, "Automated network design—The frequency domain case," *IEEE Trans, Circuit Theory*, vol. 16, pp. 330–337, Aug. 1969.
[89] W. Nye, D. Riley, A. Sangiovanni-Vincentelli, and A. Tits, "DELIGHT.SPICE: An optimization-based system for the design of integrated circuits," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 501–518, Apr. 1988.
[90] F. Medeiro *et al.*, "A statistical optimization-based approach for automated sizing of analog cells," in *Proc. ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, 1994, pp. 594–597.
[91] E. Ochotta, R. Rutenbar, and L. R. Carley, "Synthesis of high-performance analog circuits in ASTRX/OBLX," *IEEE Trans. Computer-Aided Design*, vol. 15, pp. 273–294, Mar. 1996.
[92] E. Ochotta, T. Mukherjee, R. Rutenbar, and L. R. Carley, *Practical Synthesis of High-Performance Analog Circuits*. Norwell, MA: Kluwer, 1998.
[93] R. Schwencker, F. Schenkel, H. Graeb, and K. Antreich, "The generalized boundary curve—A common method for automatic nominal design and design centering of analog circuits," in *Proc. IEEE Design Automation and Test in Europe Conf. (DATE)*, 2000, pp. 42–47.
[94] R. Phelps, M. Krasnicki, R. Rutenbar, L. R. Carley, and J. Hellums, "ANACONDA: Robust synthesis of analog circuits via stochastic pattern search," in *Proc. Custom Integrated Circuits Conf. (CICC)*, 1999, pp. 567–570.
[95] M. Krasnicki, R. Phelps, R. Rutenbar, and L. R. Carley, "MAELSTROM: Efficient simulation-based synthesis for custom analog cells," in *Proc. ACM/IEEE Design Automation Conf. (DAC)*, 1999, pp. 945–950.
[96] R. Phelps, M. Krasnicki, R. Rutenbar, L. R. Carley, and J. Hellums, "A case study of synthesis for industrial-scale analog IP: Redesign of the equalizer/filter frontend for an ADSL CODEC," in *Proc. ACM/IEEE Design Automation Conf. (DAC)*, 2000, pp. 1–6.
[97] H. Onodera, H. Kanbara, and K. Tamaru, "Operational-amplifier compilation with performance optimization," *IEEE J. Solid-State Circuits*, vol. 25, pp. 466–473, Apr. 1990.
[98] J. Crols, S. Donnay, M. Steyaert, and G. Gielen, "A high-level design and optimization tool for analog RF receiver front-ends," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1995, pp. 550–553.
[99] J. Assael, P. Senn, and M. Tawfik, "A switched-capacitor filter silicon compiler," *IEEE J. Solid-State Circuits*, vol. 23, pp. 166–174, Feb. 1988.
[100] G. Gielen and J. Franca, "CAD tools for data converter design: An overview," *IEEE Trans. Circuits Systems II:*, vol. 43, pp. 77–89, Feb. 1996.
[101] S. Director, W. Maly, and A. Strojwas, *VLSI Design for Manufacturing: Yield Enhancement*. Norwell, MA: Kluwer, 1990.
[102] J. Zhang and M. Styblinski, *Yield and Variability Optimization of Integrated Circuits*. Norwell, MA: Kluwer, 1995.
[103] T. Mukherjee, L. R. Carley, and R. Rutenbar, "Synthesis of manufacturable analog circuits," in *Proc. ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1995, pp. 586–593.
[104] G. Debyser and G. Gielen, "Efficient analog circuit synthesis with simultaneous yield and robustness optimization," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1998, pp. 308–311.
[105] J. Kuhn, "Analog module generators for silicon compilation," in *VLSI System Design*, 1987.
[106] J. Rijmenants *et al.*, "ILAC: An automated layout tool for analog CMOS circuits," *IEEE J. Solid-State Circuits*, vol. 24, pp. 417–425, Apr. 1989.
[107] J. Cohn, D. Garrod, R. Rutenbar, and L. R. Carley, *Analog Device-Level Layout Generation*. Norwell, MA: Kluwer, 1994.
[108] D. Garrod, R. Rutenbar, and L. R. Carley, "Automatic layout of custom analog cells in ANAGRAM," in *Proc. ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1988, pp. 544–547.
[109] J. Cohn, D. Garrod, R. Rutenbar, and L. R. Carley, "KOAN/ANAGRAM II: New tools for device-level analog placement and routing," *IEEE J. Solid-State Circuits*, vol. 26, pp. 330–342, Mar. 1991.
[110] M. Mogaki *et al.*, "LADIES: An automated layout system for analog LSI's," in *Proc. ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1989, pp. 450–453.
[111] V. Meyer zu Bexten, C. Moraga, R. Klinke, W. Brockherde, and K. Hess, "ALSYN: Flexible rule-based layout synthesis for analog IC's," *IEEE J. Solid-State Circuits*, vol. 28, pp. 261–268, Mar. 1993.
[112] E. Malavasi, E. Charbon, E. Felt, and A. Sangiovanni-Vincentelli, "Automation of IC layout with analog constraints," *IEEE Trans. Computer-Aided Design*, vol. 15, pp. 923–942, Aug. 1996.
[113] E. Malavasi and A. Sangiovanni-Vincentelli, "Area routing for analog layout," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 1186–1197, Aug. 1993.
[114] E. Charbon, E. Malavasi, U. Choudhury, A. Casotto, and A. Sangiovanni-Vincentelli, "A constraint-driven placement methodology for analog integrated circuits," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, May 1992, pp. 28.2.1–28.2.4.
[115] E. Malavasi, E. Felt, E. Charbon, and A. Sangiovanni-Vincentelli, "Symbolic compaction with analog constraints," *Wiley Int. Journal Circuit Theory Applicat.*, vol. 23, pp. 433–452, July–Aug. 1995.
[116] B. Basaran, R. Rutenbar, and L. R. Carley, "Latchup-aware placement and parasitic-bounded routing of custom analog cells," in *Proc. ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1993.
[117] K. Lampaert, G. Gielen, and W. Sansen, "A performance-driven placement tool for analog integrated circuits," *IEEE J. Solid-State Circuits*, vol. 30, pp. 773–780, July 1995.
[118] ——, "Analog routing for performance and manufacturability," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, May 1996, pp. 175–178.
[119] U. Choudhury and A. Sangiovanni-Vincentelli, "Automatic generation of parasitic constraints for performance-constrained physical design of analog circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 208–224, Feb. 1993.
[120] E. Charbon, E. Malavasi, and A. Sangiovanni-Vincentelli, "Generalized constraint generation for analog circuit design," in *Proc. ACM/EEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1993, pp. 408–414.
[121] E. Malavasi and D. Pandini, "Optimum CMOS stack generation with analog constraints," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 107–122, Jan. 1995.
[122] B. Basaran and R. Rutenbar, "An O(n) algorithm for transistor stacking with performance constraints," in *Proc. ACM/IEEE Design Automation Conf. (DAC)*, June 1996.
[123] J. Prieto, A. Rueda, J. Quintana, and J. Huertas, "A performance-driven placement algorithm with simultaneous place&route optimization for analog IC's," in *Proc. IEEE Eur. Design Test Conf. (ED&TC)*, 1997, pp. 389–394.
[124] R. Okuda, T. Sato, H. Onodera, and K. Tamuru, "An efficient algorithm for layout compaction problem with symmetry constraints," in *Proc. ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1989, pp. 148–151.
[125] J. Cohn, D. Garrod, R. Rutenbar, and L. R. Carley, "Techniques for simultaneous placement and routing of custom analog cells in KOAN/ANAGRAM II," in *Proc. ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1991, pp. 394–397.
[126] H. Yaghutiel, A. Sangiovanni-Vincentelli, and P. Gray, "A methodology for automated layout of switched-capacitor filters," in *Proc. ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1986, pp. 444–447.
[127] C. Kimble *et al.*, "Analog autorouted VLSI," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, June 1985.
[128] R. Gyurcsik and J. Jeen, "A generalized approach to routing mixed analog and digital signal nets in a channel," *IEEE J. Solid-State Circuits*, vol. 24, pp. 436–442, Apr. 1989.
[129] U. Choudhury and A. Sangiovanni-Vincentelli, "Constraint-based channel routing for analog and mixed analog/digital circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 497–510, Apr. 1993.
[130] S. Mitra, S. Nag, R. Rutenbar, and L. R. Carley, "System-level routing of mixed-signal ASICs in WREN," in *ACM/IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1992.
[131] S. Mitra, R. Rutenbar, L. R. Carley, and D. Allstot, "Substrate-aware mixed-signal macrocell placement in WRIGHT," *IEEE J. Solid-State Circuits*, vol. 30, pp. 269–278, Mar. 1995.
[132] B. Stanisic, N. Verghese, R. Rutenbar, L. R. Carley, and D. Allstot, "Addressing substrate coupling in mixed-mode ICs: Simulation and power distribution synthesis," *IEEE J. Solid-State Circuits*, vol. 29, Mar. 1994.
[133] B. Stanisic, R. Rutenbar, and L. R. Carley, *Synthesis of Power Distribution to Manage Signal Integrity in Mixed-Signal ICs*. Norwell, MA: Kluwer, 1996.

[134] S. Director, P. Feldmann, and K. Krishna, "Optimization of parametric yield: A tutorial," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 1992, pp. 3.1.1–3.1.8.
[135] K. Antreich, H. Graeb, and C. Wieser, "Circuit analysis and optimization driven by worst-case distances," *IEEE Trans. Computer-Aided Design*, vol. 13, pp. 57–71, Jan. 1994.
[136] C. Guardiani, P. Scandolara, J. Benkoski, and G. Nicollini, "Yield optimization of analog IC's using two-step analytic modeling methods," *IEEE J. Solid-State Circuits*, vol. 28, pp. 778–783, July 1993.
[137] M. Conti, P. Crippa, S. Orcioni, and C. Turchetti, "Parametric yield formulation of MOS IC's affected by mismatch effect," *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 582–596, May 1999.
[138] J. Huertas, "Test and design for testability of analog and mixed-signal integrated circuits," in *Selected Topics in Circuits and Systems*, H. Dedieu, Ed. Amsterdam, The Netherlands: Elsevier, 1993, pp. 77–156.
[139] B. Vinnakota, Ed., *Analog and Mixed-Signal Test*. Englewood Cliffs, NJ: Prentice-Hall, 1998.
[140] M. Sachdev, *Defect Oriented Testing for CMOS Analog and Digital Circuits*. Norwell, MA: Kluwer, 1998.
[141] C. Sebeke, J. Teixeira, and M. Ohletz, "Automatic fault extraction and simulation of layout realistic faults for integrated analogue circuits," in *Proc. IEEE Eur. Design Test Conf. (ED&TC)*, 1995, pp. 464–468.
[142] G. Devarayanadurg and M. Soma, "Dynamic test signal design for analog ICs," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 1995, pp. 627–630.
[143] W. Verhaegen, G. Van der Plas, and G. Gielen, "Automated test pattern generation for analog integrated circuits," in *Proc. IEEE VLSI Test Symp. (VTS)*, 1997, pp. 296–301.
[144] K. Wagner and T. Williams, "Design for testability of mixed signal integrated circuits," in *Proc. IEEE Int. Test Conf. (ITC)*, 1988, pp. 823–829.
[145] C. Wey, "Built-in self-test (BIST) structure for analog circuit fault diagnosis," *IEEE Trans. Instrum. Meas.*, vol. 39, pp. 517–521, June 1990.
[146] K. Arabi and B. Kaminska, "Oscillation-test strategy for analog and mixed-signal integrated circuits," in *Proc. IEEE VLSI Test Symp. (VTS)*, 1996, pp. 476–482.
[147] G. Roberts and A. Lu, *Analog Signal Generation for Built-In Self-Test of Mixed-Signal Integrated Circuits*. Norwell, MA: Kluwer, 1995.
[148] M. Ohletz, "Hybrid built-in self test (HBIST) for mixed analogue/digital integrated circuits," in *Proc. IEEE Eur. Test Conf. (ETC)*, 1991, pp. 307–316.
[149] *IEEE 1149.4 Standard for Mixed-Signal Test Bus*, 1997.

Georges G. E. Gielen (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical engineering from the Katholieke Universiteit Leuven, Leuven, Belgium, in 1986 and 1990, respectively.

In 1990, he was appointed as a Postdoctoral Research Assistant and Visiting Lecturer at the Department of Electrical Engineering and Computer Science, University of California, Berkeley. From 1991 to 1993, he was a Postdoctoral Research Assistant of the Belgian National Fund of Scientific Research at the ESAT-MICAS Laboratory of the Katholieke Universiteit Leuven. In 1993, he was appointed as a Tenure Research Associate of the Belgian National Fund of Scientific Research and, at the same time, as an Assistant Professor at the Katholieke Universiteit Leuven. In 1995, he was promoted to Associate Professor and in 2000 to full-time Professor at the same university. His research interests are in the design of analog and mixed-signal integrated circuits and especially in analog and mixed-signal CAD tools and design automation (modeling, simulation and symbolic analysis, analog synthesis, analog layout generation, analog and mixed-signal testing). He is the Coordinator or Partner of several (industrial) research projects in this area. He has authored or coauthored two books and more than 100 papers in edited books, international journals, and conference proceedings.

Dr. Gielen was the 1997 Laureate of the Belgian Royal Academy of Sciences, Literature and Arts, in the category of engineering sciences. He also received the 1995 Best Paper Award from the John Wiley international journal on *Circuit Theory and Applications*. He is a regular member of the Program Committees of international conferences (ICCAD, DATE, CICC, etc.), he is an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, PART II, and is a Member of the Editorial Board of the Kluwer international journal on *Analog Integrated Circuits and Signal Processing*. He is a Member of the Board of Governors of the IEEE Circuits and Systems (CAS) Society and is the Chairman of the IEEE Benelux CA CAS Chapter.

Rob. A. Rutenbar (Fellow, IEEE) received the Ph.D. degree from the University of Michigan, Ann Arbor, in 1984.

He subsequently joined the faculty of Carnegie Mellon University (CMU), Pittsburgh, PA. He is currently Professor of Electrical and Computer Engineering, and (by courtesy) of Computer Science. From 1993 to 1998, he was Director of the CMU Center for Electronic Design Automation. He is Cofounder of NeoLinear, Inc., and served as its Chief Technologist on a 1998 leave from CMU. His research interests focus on circuit and layout synthesis algorithms for mixed-signal ASICs, for high-speed digital systems, and for FPGAs.

Dr. Rutenbar received a Presidential Young Investigator Award from the National Science Foundation in 1987. He has won Best/Distinguished paper awards from the Design Automation Conference (1987) and the International Conference on CAD (1991). He has been on the program committees for the IEEE International Symposium on FPGAs, and the ACM International Symposium in Physical Design. He also served on the Editorial Board of IEEE Spectrum. He was General Chair of the 1996 ICCAD. He Chaired the Analog Technical Advisory Board for Cadence Design Systems from 1992 through 1996. He is a Member of the ACM and Eta Kappa Nu.