Synthesis Tools for Mixed-Signal ICs: Progress on Frontend and Backend Strategies

L. Richard Carley*, Georges G.E. Gielen, Rob A. Rutenbar* and Willy M.C. Sansen

*Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213 USA

Electrical Engineering
Katholieke Universiteit Leuven
Leuven, Belgium

Abstract

Digital synthesis tools such as logic synthesis and semicustom layout have dramatically changed both the *frontend* (specification to netlist) and *backend* (netlist to mask) steps of the digital IC design process. In this tutorial, we look at the last decade's worth of progress on analog circuit synthesis and layout tools. We focus on the frontend and backend of analog and mixed-signal IC design flows. The tutorial summarizes the problems for which viable solutions are emerging, and those which are still unsolved.

1 Introduction

The microelectronics market and in particular the markets for ASICs, ASSPs and high-volume commodity ICs are characterized by an ever increasing level of integration complexity. In recent years, complete systems that before occupied one or more boards are increasingly being integrated on a few chips or even one single chip. Examples of such "systems on a chip" are the single-chip TV or the single-chip camera, as presented at the 1996 International Solid-State Circuits Conference. Although most functions in such integrated systems are implemented with digital or 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. The booming market share of mixed-signal ASICs in modern electronic systems for telecom, consumer, computing, and automotive applications is a direct result of this. Since the early 1990s, the average growth rate of the mixed-signal IC market has been between 15 and 20% per year.

Together with this increase in circuit complexity, also the design complexity has increased drastically. At the same time, many present ASIC application markets are characterized by shortening product life cycles and time-to-market constraints. These constraints can only be met by using advanced computer-aided design (CAD) tools. In the digital world, logic synthesis and semi-custom layout have emerged as the *de facto* strategies for managing the *frontend* (specification to gate-level netlist steps) and the *backend* (netlist to mask steps) of the design process. Unfortunately, we do not (yet) have robust circuit synthesis and layout tools in the analog domain. The design cycle for analog and mixed-signal ICs remains long and error-prone.

This tutorial attempts to address this lack of mature, commercial analog CAD tools. Our central purpose is to suggest that the news is *not* all bad here: the circuits research community has been working aggressively for over a decade to solve the difficult design problems posed by analog and mixed-signal ICs. Given the success of synthesis and semi-custom layout in digital designs, we focus our tutorial also on tools to support the *frontend* and *backend* of the analog design process, *i.e.*, custom circuit synthesis and layout. Given space limitations, this necessarily means that our review of relevant efforts in each area is brief and some interesting work is omitted. Two omissions we note specifically. We do not treat any analog hardware description behavioral modeling languages. The topic is extremely important and is likely to have a substantial impact on how analog synthesis moves forward, but as of this writing the dust has not yet settled on any of the proposed standards. And, we do not cover analog and mixed-signal simulation strategies. These are much more mature than the comparable synthesis and layout tools, and there are numerous commercial offerings. With those caveats, this tutorial strives to give an overview of the state-of-the-art in the field of analog circuit and layout synthesis, where we are today, and where we are going tomorrow.

2 Frontend Solutions: Analog Synthesis

In this section we present the hierarchical design methodology used in most analog CAD systems today, followed by a survey of the different approaches undertaken towards analog circuit synthesis. This frontend synthesis consists of two steps: topology selection and circuit sizing.

2.1 Hierarchical Design Methodology

For the design of a complex analog macroblock like 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), each of which is then designed separately. In such a hierarchical scheme, most experimental analog CAD systems presented today use a performance-driven design strategy, that consists of the alternation of the following steps in between two levels of the design hierarchy [1,2,3]:

- **Top-down path:**
  - topology selection
  - specification translation (circuit sizing)
  - design verification
- **Bottom-up path:**
  - layout generation
  - detailed design verification (after extraction)

Throughout the design constraints have to be passed down the hierarchy in order to make sure that the top-level block at the end meets its specifications. Redesign iterations are needed when the design fails to meet the specifications at some point in the design

33rd Design Automation Conference ®

DAC 96 - 06/96 Las Vegas, NV, USA ©1996 ACM, Inc. 0-89791-833-9/96/0006.. $3.50

flow.

In this design process topology selection is the step of selecting the most appropriate circuit topology out of a set of alternatives, that can best meet the given specifications. A topology can be defined hierarchically in terms of lower-level subblocks. For an analog-to-digital converter for instance, this could be selecting between a flash, a successive approximation, a Delta-Sigma or any other topology. Specification translation is then the step of mapping the specifications for the block under design at a given level (e.g. converter) into individual specifications for each of the subblocks (e.g. comparator) within the selected block topology, so that the complete block meets its specifications, while possibly also optimizing the design towards some application-specific design objectives (e.g. minimal power consumption). The translated specifications can then be verified by means of (behavioral or circuit) simulations. At the lowest level in the design hierarchy, the subblocks are single devices and specification translation reduces to circuit sizing (or dimensioning), which is the determination of all bias parameters, element values and device sizes in the circuit tuned to each specific application.

2.2 Approaches Towards Analog Circuit Synthesis

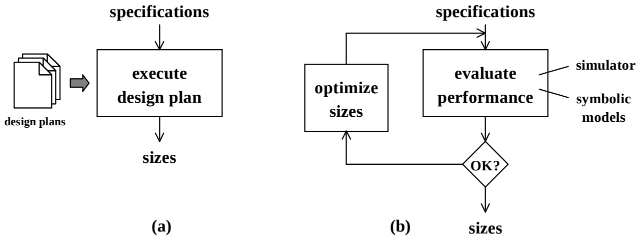
Circuit synthesis is the inverse operation of circuit analysis, where the subblock parameters (such as device sizes and bias values) are given and the resulting performance of the overall block is calculated, as is done in SPICE. During synthesis, the block performance is specified and values for the subblock parameters needed to meet these specifications have to be determined. This inverse process is not a one-to-one mapping, but usually is an underconstrained problem with many degrees of freedom. The different analog circuit synthesis systems that have been explored up till now can essentially be classified in the way how they eliminate these degrees of freedom.

The first class of analog synthesis systems presented in the mid to late eighties were *knowledge-based*. Specific heuristic design knowledge about the circuit topology under design was encoded explicitly in some computer executable form, that was then executed during the synthesis run for a given set of input specifications to obtain the design solution. The knowledge was encoded in different ways in different systems. The IDAC tool [4] 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. This approach is illustrated schematically in Fig. 1a.

The big advantage of using design plans is their fast execution speed, which allows for fast performance space explorations [1]. The big disadvantages are the lack of flexibility and the large time needed to develop a 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 [5] that the creation of a design script or plan typically takes 4 times more effort than is needed to actually design the circuit once. Considering the large number of circuit schematics in use in industrial practice, this essentially restricted the commercial usability of the 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 did not fundamentally change this [6]. The selection of the topology had to be carried out by the designer himself in IDAC.

OASYS [1] adopted a similar design plan based sizing approach, but explicitly introduced hierarchy in the design of analog circuits and also added a heuristic approach towards topology selection to the system. 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. Collecting and ordering all the design knowledge in the design plan however still remained a time-consuming job. The approach was later on adopted in the commercial MIDAS system [5]. Other ways to encode the knowledge have been explored as well, such as in BLADES which is a rule-based system to size analog circuits.

Figure 1. The two basic approaches towards analog circuit synthesis: a) knowledge-based, b) optimization-based.



In order to make analog design systems much more open for new circuit schematics, an alternative solution was sought since the late eighties in using 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. This approach is illustrated schematically in Fig. 1b. At each iteration of the optimization routine, the performance of the circuit has to be evaluated. Depending on which method is used for this, two different subcategories can be distinguished.

In the subcategory of *equation-based optimization approaches* (simplified) analytic design equations are used to describe the circuit performance. In OPASYN [8] and later also CADICS [9] the design equations still had to be derived and ordered by hand, but the degrees of freedom were resolved implicitly by optimization. The tool performed rule-based topology selection. The OPTIMAN program [10,11] added the use of a global simulated annealing algorithm, but also tried to solve two remaining problems. The symbolic simulator ISAAC [12] was developed to automatically generate the (simplified) design equations needed to evaluate the circuit performance and in this way to reduce the introduction time for new circuit schematics. Computer-aided symbolic analysis is now possible for the ac behavior (both linear and weakly nonlinear) of analog circuits up to the complexity of an entire 741 opamp. The symbolic equations can also be used to provide a designer insight into the behavior of an analog circuit. The second problem of ordering the design equations into an application-specific design or evaluation plan was then tackled using constraint programming techniques in the DONALD program [13,14]. Together with a separate topology selection tool based on boundary checking and interval analysis [15], all these tools are now integrated into the AMGIE analog circuit synthesis system from K.U. Leuven [16]. Table 1 shows the results of a recent synthesis experiment for a pulse detector frontend (consisting of a charge-sensitive amplifier and a 4-stage pulse-shaping amplifier). A reduction of the power consumption with a factor of 6 was achieved by the synthesis system compared to the solution generated by an expert designer.

Several other equation-based tools exist as well, such are for instance STAIC [17] and ISAID [18]. An application of this technique to the high-level synthesis of data converters can be found in AZTECA/CATALYST [19] and SDOPT [20], the two tools targeting different types of converters.

Table 1. Example of synthesis experiment.

performance	**specification**	**manual**	**synthesis**
peaking time	< 1.5 µs	1.1 µs	1.1 µs
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

A drawback of the equation-based approach is that the design equations still have to be derived, which for certain characteristics such as transient or large-signal responses can be quite tedious to do with sufficient accuracy. Therefore, in recent years and with improving computer power, a second subcategory of approaches has been presented that perform a *full SPICE simulation* run at every iteration of the optimization. For a limited set of parameters this was already possible in DELIGHT.SPICE [21]. However, the challenge was to solve for all degrees of freedom, in case no good initial starting point could be provided. The FRIDGE tool [22] calls the SPICE simulator throughout a simulated annealing optimization loop, and is in this way capable of synthesizing low-level analog circuits (e.g. opamps). The introduction of a new circuit schematic in such an approach is relatively easy, but the drawback are the long run times, especially if the initial search space is large.

An in-between solution was therefore explored in the ASTRX/OBLX tool from CMU [23], where the linear small-signal characteristics are simulated efficiently using AWE [61], whereas equations have to be provided for all other characteristics. ASTRX compiles the initial synthesis specification into an executable cost function whose minimum represents a good solution; OBLX then numerically searches for a good minimum of this function via annealing. For efficiency, the tool also uses a dc-free biasing formulation of the analog design problem, where the dc constraints are solved by relaxation throughout the optimization run. ASTRX/OBLX has been successful in a wide variety of cell-level designs [24]. Other simulation-based approaches can be found in tools such as OAC [25], which is based on redesign starting from a previous design solution stored in the system's database.

Other tools have attempted to integrate the topology selection step as part of the optimization loop. This was done using a mixed optimization formulation with boolean variables representing topological choices [26], or by using a genetic algorithm to find the best topology choice [27,28].

A recent application of the simulation-based optimization approach to the high-level optimization of analog RF receiver frontends was presented in [29]. A dedicated RF front-end simulator was developed and used to calculate the ratio of the wanted signal to all kinds of unwanted signals (noise, distortion, aliasing...) in the frequency band of interest. An optimization 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 describe the different subblocks at this high level.

In summary, the initial design systems like IDAC were too closed and therefore failed on the market place. The trend towards 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 has been evident. The research progress over the last ten years has resulted in the development of several experimental analog synthesis systems, with which several designs have successfully been synthesized, fabricated and measured; this includes not only operational amplifiers but also filters [30] and data converters.

Finally, it has to be added that industrial design practice not only cares for a fully optimized nominal design solution, 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. Precautions for this were already hardcoded in the design plans of IDAC [4], but are more difficult to incorporate in optimization-based approaches. The ASTRX/OBLX tool has been extended with these manufacturability considerations [31]; the strategy uses a nonlinear infinite programming formulation to search for the worst-case “corners” at which the evolving circuit should be evaluated for correct performance. The approach has been successful in several test cases but does increase the CPU time required (e.g., by roughly 4X-10X). More work in this direction is clearly needed.

3 Backend: Analog Cell and System Layout

Perhaps unsurprisingly, analog layout is a bit more mature than analog circuit synthesis, in large part because it has been able to leverage mature ideas from digital IC layout. In our review we distinguish two different layout problems: *cell layout*, which transforms a transistor-level schematic with 10-100 devices into a mask layout, and *system assembly*, in which large, basic functional blocks are laid out, and the goal is to floorplan, place, and route them.

3.1 Cell Layout Strategies

The earliest approaches to custom analog cell layout relied on procedural module generation. These approaches are a workable strategy when the analog cells to be laid out are relatively static, *i.e.*, necessary changes in device sizing or biasing result in little need for global alterations in device layout, orientation, reshaping, *etc*. In such instances, a procedural generation scheme which starts with a basic geometric template and completes it by correctly sizing the devices and wires can be quite satisfactory. [32] is an often cited early example. The more recent system at Philips [5] is a good example of practical application of these ideas on complex circuits.

Often, however, changes in circuit design require full custom layout, which can be handled with a *macrocell-style* strategy. The terminology is borrowed from digital floorplanning algorithms, which manipulate flexible layout blocks, arrange them topologically, and then route them. For analog cells, we regard the flexible blocks as devices to be reshaped and reoriented as necessary. Module generation techniques are used to generate the layouts of the individual devices. A placer then arranges these devices, and a router interconnects them—all while attending to the numerous parasitics and couplings to which analog circuits are (unfortunately) sensitive.

ILAC from CSEM was an important first attempt in this style [33]. It 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 by compaction. 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.

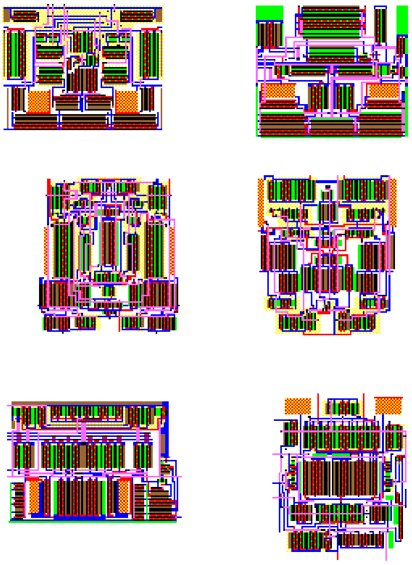
ANAGRAM and its successor KOAN / ANAGRAM II from CMU kept the macrocell style, but reinvented the necessary algorithms from the bottom up, incorporating many manual design optimizations [34,35,36]. For example, the device placer KOAN relied on a very small library of device generators, and migrated important layout optimizations into the placer itself. KOAN could dynamically fold, merge and abut MOS devices, and thus discover desirable optimizations to minimize parasitic capacitance during placement. KOAN was based on an efficient simulated annealing algorithm. 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.

Other device placers and routers operating in the *macrocell-style* have appeared (e.g., [37,38]), confirming its utility. Results from tools using this approach can be quite impressive. For example, Figure 2 (from [36]) shows six analog cell layouts—four manual and two from KOAN/ANAGRAM II. The automatic layouts compare favorably to the manual ones.

In the next generation of cell-level tools, the focus shifted to quantitative optimization of performance goals. For example, KOAN maximized MOS drain-source merging during layout, and ANAGRAM II minimized crosstalk, but without any specific, quantitative performance targets. The routers ROAD [39] and ANAGRAM III [40] use improved cost-based schemes that route instead to minimize the deviation from acceptable parasitic bounds derived from designers or sensitivity analysis. The router in [41] can manage not just parasitic sensitivities, but also basic yield and testability concerns. Similarly, the placer in [42] augments a KOAN-style model with sensitivity analysis so that performance degradations due to layout parasitics can be accurately controlled.

In the newest generation of CMOS analog cell layout tools, the device placement task has been separated into two distinct phases: 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. [43] gave an exact algorithm to extract all the optimal stacks, and the placer in [44] extends a KOAN-style algorithm to dynamically choose the right stacking and the right placement of each stack. [45] offers another variant of this idea: instead of extracting all the stacks (which can be time-consuming since the underlying algorithm is exponential), this technique 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.

Figure 2. ***KOAN/ANAGRAM II Cell Layouts.*** **Six layouts of the identical CMOS opamp are shown. The two middle layouts are automatic, the rest manual.**



The notion of using sensitivity analysis to quantify the impact on final circuit performance of low-level layout decisions (*e.g.*, device merging, symmetric placement / routing, parasitic coupling due to specific proximities, *etc.*) has emerged as the critical glue that links the various approaches being taken for cell level layout and system assembly. Several systems from U.C. Berkeley are notable here. An influential early formulation of the sensitivity analysis problem was [46] which not only quantified layout impacts on circuit performance, but also showed how to use nonlinear programming techniques to map these sensitivities into constraints on various portions of the layout task. In related work, [47] showed how to extract critical constraints on symmetry and matching directly from a device schematic.

One final problem in the macrocell style is the separation of the placement and routing steps. In manual cell layout, there is no effective difference between a rectangle representing a wire and one representing part of a device: they can each be manipulated simultaneously. In a place-then-route strategy, one problem is estimating how much space to leave around each device for the wires. One solution strategy is *analog compaction*, *e.g.*, [48,49], in which we leave extra space during device placement and then compact. A more radical alternative is *simultaneous device place-and-route*. An experimental version of KOAN [50] supported this by iteratively perturbing both the wires and the devices.

We expect to see *macrocell-style* custom cell layout schemes maturing over the next few years. Of course, there are still problems to solve. The wirespace problem remains, though simultaneous place-and-route seems to offer an elegant solution. As we noted in [51] an open problem is “closing the loop” from cell synthesis to cell layout, so that layouts which do not meet specifications can, if necessary, cause actual circuit design changes (via circuit resynthesis). How to control this loop, and how to reflect layout concerns in synthesis and synthesis concerns in layout remain difficult.

3.2 Mixed-Signal System Assembly and Layout

A mixed-signal system is a set of custom analog and digital functional blocks. *Assembly* means floorplanning, placement, global and detailed routing (including the power grid). As well as parasitic sensitivities, the new problem at the chip level is coupling between digital switching noise and sensitive analog circuits.

Just as at the cell level, procedural generation remains a viable alternative for well-understood designs with substantial regularity (*e.g.*, switched capacitor filters [52], or data converters [5]).

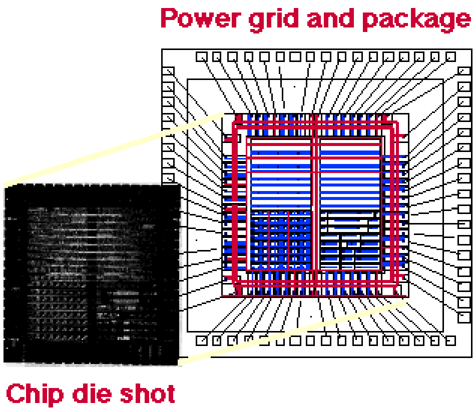
More generally though, work has focussed on custom placement and routing at the block level. For row-based layout, an early elegant solution to the coupling problem was the *segregated channels* idea of [53] to alternate noisy digital and sensitive analog wiring channels in a row-based cell layout. The 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 [54], it was observed that a well-known digital channel routing algorithm could be easily extended to handle critical analog problems that involve varying wire widths and wire separations needed to isolate interacting signals. Work at Berkeley substantially extended this strategy to handle complex analog symmetries, and the insertion of shields between incompatible signals [55].

The WREN [56] and WRIGHT [57] systems from CMU generalized these ideas to the case of arbitrary layouts of mixed functional blocks. WREN comprises both a mixed-signal global router and channel router. WREN introduced the notion of *SNR-style* (signal-to-noise ratio) 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 (influenced by [46]) 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 (as in [55]). WRIGHT uses a KOAN-style annealer to floorplan the blocks, but with a fast substrate noise coupling evaluator so that a simplified view of substrate noise influences the floorplan. (We should mention here that substrate coupling is an increasingly difficult problem as more and faster digital logic is placed side-by-side with sensitive analog parts. See [58,59] for detailed treatments on substrate coupling.)

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 (non-tree) 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 from CMU [58,60] addresses these concerns by casting mixed-signal power grid synthesis as a routing problem that uses fast AWE-based [61] linear system evaluation to electrically model the entire power grid, package and substrate during layout. Figure 3. shows an example RAIL redesign of the data channel from [62] in which a demanding set of dc, ac and transient performance constraints were met automatically.

Most of these system layout tools are fairly recent, but because they often rely on mature core algorithms from similar digital layout problems, many have been prototyped both successfully and quickly. Several full, top-to-bottom prototypes have recently emerged (*e.g.* [63,64]). We believe 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, and to be practical for practicing designers.

Figure 3. RAIL power grid design for IBM data channel



4 Conclusions

Despite the dearth of commercial offerings, there has been substantial progress on tools for custom analog circuit synthesis and layout over the last decade. 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 are beginning to show glimmers of practical application. We are not “there” yet, but we are making real progress.

Acknowledgments

We are grateful to our many students and colleagues at CMU and K.U. Leuven for their assistance with the preparation of this manuscript. The authors at CMU were supported by the Semiconductor Research Corp, U.S. National Science Foundation, IBM, Intel and Harris. The authors at KUL acknowledge support from several industrial companies, E.C. ESPRIT, Belgian IUAP, ESA-ESTEC.

References

[1] R. Harjani, R. Rutenbar, L. Carley, “OASYS: a framework for analog circuit synthesis,” *IEEE Trans. CAD*, Vol. 8, No. 12, pp. 1247-1266, December 1989.

[2] G. Gielen, K. Swings, W. Sansen, “Open analog synthesis system based on declarative models,” chapter 18 in *Analog Circuit Design* (J. Huijsing, R. van der Plassche and W. Sansen. eds.), Kluwer Academic Publishers, pp. 421-445, 1993.

[3] E. Malavasi *et al.*, “A top-down, constraint-driven design methodology for analog integrated circuits,” chapter 13 in *Analog Circuit Design* (J. Huijsing, R. van der Plassche and W. Sansen. eds.), Kluwer Academic Publishers, pp. 285-324, 1993.

[4] M. Degrauwe, *et al.*, “IDAC: an interactive design tool for analog CMOS circuits,” *IEEE JSSC*, Vol. 22, No. 6, pp. 1106- 1116, December 1987.

[5] G. Beenker, J. Conway, G. Schrooten, A. Slenter, “Analog CAD for consumer ICs,” chapter 15 in “Analog circuit design” *Analog Circuit Design* (J. Huijsing, R. van der Plassche and W. Sansen. eds.), Kluwer Academic Publishers, pp. 347- 367, 1993.

[6] R. Henderson, *et al.*, “A spreadsheet interface for analog design knowledge capture and re-use,” *Proc. IEEE CICC*, 13.3, May 1993.

[7] F. El-Turky, E. Perry, “BLADES: an artificial intelligence approach to analog circuit design,” *IEEE Trans. CAD*, Vol. 8, No. 6, pp. 680-692, June 1989.

[8] H. Koh, C. Séquin, P. Gray, “OPASYN: a compiler for CMOS operational amplifiers,” *IEEE Trans. CAD*, Vol. 9, No. 2, pp. 113-125, February 1990.

[9] G. Jusuf, P. Gray, A. Sangiovanni-Vincentelli, “CADICS - Cyclic analog-to- digital converter synthesis,” *Proc. ACM/IEEE ICCAD*, pp. 286-289, 1990.

[10] G. Gielen, H. Walscharts, W. Sansen, “Analog circuit design optimization based on symbolic simulation and simulated annealing,” *IEEE JSSC*, Vol. 25, No. 3, pp. 707-713, June 1990.

[11] G. Gielen, W. Sansen, *Symbolic Analysis For Automated Design Of Analog Integrated Circuits*, Kluwer Academic Publishers, 1991.

[12] G. Gielen, H. Walscharts, W. Sansen, “ISAAC: a symbolic simulator for analog integrated circuits,” *IEEE JSSC*, Vol. 24, No. 6, pp. 1587-1597, Dec. 1989.

[13] K. Swings, W. Sansen, “DONALD: a workbench for interactive design space exploration and sizing of analog circuits,” *Proc. EDAC*, pp. 475-479, 1991.

[14] K. Swings, W. Sansen, “ARIADNE: a constraint-based approach to computer-aided synthesis and modeling of analog integrated circuits”, *Kluwer International Journal on Analog Integrated Circuits and Signal Processing*, Vol. 3, pp. 197-215, 1993.

[15] P. Veselinovic, *et al.*, “A flexible topology selection program as part of an analog synthesis system,” *Proc. ED&TC*, pp. 119-123, 1995.

[16]G. Gielen, *et al.*, “An analog module generator for mixed analog/digital ASIC design,” *International Journal of Circuit Theory and Applications*, Vol. 23, pp. 269-283, July-August 1995.

[17]J.P. Harvey, *et al.*, “STAIC: An interactive framework for synthesizing CMOS and BiCMOS analog circuits,” *IEEE Trans. CAD*, Vol. 11, No. 11, pp. 1402-1415, Nov. 1992.

[18]C. Toumazou, C.A. Makris, “Analog IC design automation: Part I -- Automated circuit generation: New concepts and methods,” *IEEE Trans. CAD*, Vol. 14, No. 2, pp. 218-238, Feb. 1995.

[19]N. Horta, J. Franca, C. Leme, “Automated high level synthesis of data conversion systems,” chapter 17 of *Analogue-Digital Asics - Circuit Techniques*, *Design Tool And Applications* (Soin, Maloberti and Franca, eds.), Peter Peregrinus Ltd., 1991.

[20]F. Medeiro, B. Pérez-Verdú, A. Rodríguez-Vázquez, J. L. Huertas, “A vertically-integrated tool for automated design of SD modulators,” *Proc. ESSCIRC*, pp. 164- 167, 1994.

[21]W. Nye, D. Riley, A. Sangiovanni-Vincentelli, A. Tits, “DELIGHT.SPICE: an optimization-based system for the design of integrated circuits,” *IEEE Trans. CAD*, Vol. 7, No. 4, pp. 501-519, April 1988.

[22]F. Medeiro, *et al.*, “A statistical optimization-based approach for automated sizing of analog cells,” *Proc. ACM/IEEE ICCAD*, pp. 594-597, Nov. 1994.

[23]E.S. Ochotta, R.A. Rutenbar, L.R. Carley, “ASTRX/OBLX: Tools for rapid synthesis of high-performance analog circuits,” *IEEE Trans. CAD*, to appear 1996.

[24]E.S. Ochotta, L.R. Carley, R.A. Rutenbar “Analog circuit synthesis for large, realistic cells: designing a pipelined A/D converter with ASTRX/OBLX,” *Proc. IEEE CICC*, May 1994.

[25]H. Onodera, *et al.*, “Operational amplifier compilation with performance optimization,” *IEEE JSSC*, Vol. SC-25, No. 2, pp. 466-473, Apr. 1990.

[26]P.C. Maulik, L.R. Carley, R.A. Rutenbar “Simultaneous topology selection and sizing of cell-level analog circuits,” *IEEE Trans. CAD*, Vol. 14, No. 4, April 1995.

[27]Z.Q. Neng, et al., “SEAS: A simulated evolution approach for analog circuit synthesis,” *Proc. IEEE CICC*, 5.2, May 1991.

[28]W. Kruiskamp, D. Leenaerts, “DARWIN: CMOS opamp synthesis by means of a genetic algorithm,” *Proc. ACM/IEEE DAC*, pp. 433-438, June 1995.

[29]J. Crols, S. Donnay, M. Steyaert, G. Gielen, “A high-level design and optimization tool for analog RF receiver front-ends,” proc. ICCAD, pp. 550-553, 1995.

[30]J. Assael, P. Senn, M. Tawfik, “A switched-capacitor filter silicon compiler,” IEEE Journal of Solid-State Circuits, Vol. 23, No. 1, pp. 166-174, February 1988.

[31]T. Mukherjee, L.R. Carley, R.A. Rutenbar, “Synthesis of manufacturable analog circuits,” *Proc. ACM/IEEE ICCAD*, pp. 586-593, Nov. 1995.

[32]J. Kuhn, “Analog Module Generators for Silicon Compilation,” *VLSI Systems Design*, May 1987.

[33]J. Rijmenants. *et al.*, “ILAC: An automated layout tool for analog cmos circuits,” *IEEE JSSC*, Vol. 24, No. 4, pp. 417-425, April 1989.

[34]D. J. Garrod, R. A. Rutenbar, L. R. Carley, “Automatic layout of custom analog cells in ANANGRAM”, *Proc. ICCAD*, pp. 544-547, Nov. 1988.

[35]J. M. Cohn, D. J. Garrod, R. A. Rutenbar, L. R. Carley, “KOAN/ANAGRAM II: New tools for device-level analog placement and routing,” *IEEE JSSC*, Vol. 26, No. 3, March, 1991.

[36]J.M Cohn, D. J. Garrod, R. A. Rutenbar, L. R. Carley, *Analog Device-Level Layout Automation*, Kluwer Acad. Publ., 1994.

[37]M. Mogaki, *et al.*, “LADIES: An automatic layout system for analog LSI’s,” *Proc. ACM/IEEE ICCAD*, pp. 450-453, Nov. 1989.

[38]E. Malavasi, *et al.*, “A routing methodology for analog integrated circuits,” *Proc. ACM/IEEE ICCAD*, pp. 202-205, Nov 1990.

[39]E. Malavasi, A. Sangiovanni-Vincentelli, “Area routing for analog layout,” *IEEE Trans. CAD*, Vol. 12, No. 8, pp. 1186-1197, Aug, 1993.

[40]B. Basaran, R. A. Rutenbar, L. R. Carley, “Latchup-aware placement and parasitic-bounded routing of custom analog cells”, *Proc. ACM/IEEE ICCAD*, Nov. 1993.

[41]K. Lampaert, G. Gielen, W. Sansen, “Analog routing for manufacturability,” *Proc. IEEE CICC*, May 1996.

[42]K. Lampaert, G. Gielen, W.M. Sansen, “A performance-driven placement tool for analog integrated circuits,” *IEEE JSSC*, Vol. 30, No. 7, pp. 773-780, July 1995.

[43]E. Malavasi, D. Pandini, “Optimum CMOS stack generation with analog constraints”, *IEEE Trans. CAD*, Vol. 14, No. 1, pp. 107-12, Jan. 1995.

[44]E. Charbon, E. Malavasi, U. Choudhury, A. Casotto, A. Sangiovanni-Vincentelli, “A constraint-driven placement methodology for analog integrated circuits”, *Proc. IEEE CICC*, pp. 28.2/1-4, May 1992.

[45]B. Basaran, R. A. Rutenbar, “An O(n) algorithm for transistor stacking with performance constraints”, *Proc. ACM/IEEE DAC*, June 1996.

[46]U. Choudhury, A. Sangiovanni-Vincentelli, “Automatic generation of parasitic constraints for performance-constrained physical design of analog circuits”, *IEEE Trans. CAD*, Vol. 12, No. 2, pp. 208-224, February 1993.

[47]E. Charbon, E. Malavasi, A. Sangiovanni-Vincentelli, “Generalized constraint generation for analog circuit design”, *Proc. IEEE/ACM ICCAD*, pp. 408-414, Nov. 1993.

[48]R. Okuda, T. Sato, H. Onodera, K. Tamuru, “An efficient algorithm for layout compaction problem with symmetry constraints,” *Proc. IEEE ICCAD*, pp. 148-151, Nov. 1989.

[49]E. Malavasi, E. Felt, E. Charbon, A. Sangiovanni-Vincentelli, “Symbolic compaction with analog constraints,” *Int. J. Circuit Theory and Applic.*, Vol. 23, No. 4, pp. 433-452, Jul/Aug 1995.

[50]J. Cohn, D. Garrod, R. Rutenbar, L. R. Carley, “Techniques for simultaneous placement and routing of custom analog cells in KOAN/ANAGRAM II,” *Proc. ACM/IEEE ICCAD*, pp. 394-397, Nov. 1991.

[51]R.A. Rutenbar, “Analog design automation: Where are we? Where are we going?” *Proc. IEEE CICC*, May 1993.

[52]H. Yaghutiel, A. Sangiovanni-Vincentelli, P. R. Gray, “A methodology for automated layout of switched capacitor filters,” in *Proc. ACM/IEEE ICCAD*, Nov. 1986.

[53]C. D. Kimble, *et al*., “Analog autorouted VLSI,” *Proc. ACM/IEEE CICC.*, June 1985.

[54]R. S. Gyurcsik, J. C. Jeen, “A generalized approach to routing mixed analog and digital signal nets in a channel,” *IEEE JSSC*, Vol. 24, No. 2, pp. 436-442, Apr. 1989.

[55]U. Choudhury, A. Sangiovanni-Vincentelli, “Constraint-based channel routing for analog and mixed analog/digital circuits,” *IEEE Trans. CAD*, Vol. 12, No. 4, pp. 497-510, Apr. 1993.

[56]S. Mitra, S. Nag, R. A. Rutenbar, and L. R. Carley, “System-level routing of mixed-signal ASICs in WREN,” *Proc. ACM/IEEE ICCAD*, Nov. 1992.

[57]S. Mitra, R. A. Rutenbar, L. R. Carley, D.J. Allstot, “Substrate-aware mixed-signal macrocell placement in WRIGHT,” *IEEE JSSC*, Vol. 30, No. 3, pp. 269-278, Mar. 1995.

[58]B. R. Stanisic, N. K. Verghese, R. A. Rutenbar, L. R. Carley, D. J. Allstot, “Addressing substrate coupling in mixed-mode IC's: simulation and power distribution synthesis”, *IEEE JSSC*, Vol. 29, No. 3, Mar. 1994.

[59]N.K. Verghese, D.J. Allstot, T. Schmerbeck, *Simulation Techniques and Solutions for Mixed-Signal Integrated Circuits*, Kluwer Acad. Pub., 1995.

[60]B.R. Stanisic, R.A. Rutenbar, L.R. Carley, *Synthesis of Power Distribution to Manage Signal Integrity in Mixed-Signal ICs*, Kluwer Academic Publishers, 1996.

[61]L. T. Pillage, R. A. Rohrer, “Asymptotic waveform evaluation for timing analysis”, *IEEE Trans. CAD*, Vol. 9, No. 4, Apr. 1990.

[62]T. Schmerbeck, R. Richetta, L. Smith, “A 27 MHz mixed analog/digital recording channel DSP using partial response signalling with maximum likelihood detection,” in *Tech. Digest IEEE ISSCC*, Feb. 1991.

[63]R. Rutenbar *et al*., “Synthesis and layout for mixed-signal ics in the acacia system,” in *Analog Circuit Design*, (J.H. Huijsing, R. J. van de Plassche and W.M.C Sansen, eds.), Kluwer Acad. Publishers, pp. 127-146, 1996.

[64]E. Malavasi, E. Felt, E. Charbon and A. Sangiovanni-Vincentelli, “Automation of IC Layout with Analog Constraints,” *IEEE Trans. CAD*, to appear 1996.