A Performance-Driven Placement Tool for Analog Integrated Circuits

Koen Lampaert, Georges Gielen, *Member, IEEE,* and Willy M. Sansen, *Senior Member, IEEE*

***Abstract*—This paper presents a new approach toward performance-driven placement of analog integrated circuits. The freedom in placing the devices is used to control the layout-induced performance degradation within the margins imposed by the designer's specifications. This guarantees that the resulting layout will meet all specifications by construction. During each iteration of the simulated annealing algorithm, the layout-induced performance degradation is calculated from the geometrical properties of the intermediate solution. The placement tool inherently handles symmetry constraints, circuit loading effects and device mismatches. The feasibility of the approach is demonstrated with practical circuit examples.**

I. Introduction

GENERATING the layout of high-performance analog circuits is a difficult and time-consuming task which has a considerable impact on circuit performance. Asymmetries and device mismatches can easily upset the critical precision of component values and together with the parasitics associated with the interconnections they can introduce intolerable performance degradation. Since these parasitics are unavoidable, the main concern in computer-aided analog layout synthesis is to control the effects of the parasitics on circuit performance and to keep the layout-induced performance degradation within user-defined margins, as specified in the circuit's performance specifications. In this way, it can be guaranteed that the resulting layout will meet the specifications by construction.

Many existing analog layout programs, like KOAN/ANAGRAM [1] or ALSYN [2], however, try to optimize the layout without quantifying the performance degradation. They therefore cannot guarantee that the resulting layout will also meet the specifications. In recent years therefore, a performance-driven methodology has been introduced [3] which does try to achieve this. The approach in [3] and [4] first maps the performance specifications onto a set of constraints for critical parasitics which are then used to drive the layout tools (see Fig. 1(a)). Using sensitivity information and quadratic optimization, the set of constraints that maximizes the flexibility of the layout tools is computed. If the layout tools fail to meet one of these parasitic constraints, one or more iterations with another set of constraints is needed. This approach suffers from a number of drawbacks. First, the flexibility of layout tools is something which is very hard, if not impossible to quantify with any reasonable accuracy, for use as cost function in the optimization. Second, by imposing *one* set of constraints only, several valid alternative solutions are rejected, unnecessarily increasing the number of time-consuming iterations.

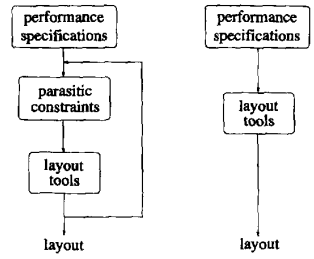


Fig. 1. Performance-driven layout methodologies: (a) with an intermediate constraint generation step, and (b) performance specifications directly taken into account by the layout tools.

To overcome these drawbacks, we therefore propose an alternative solution which eliminates the intermediate constraint generation step, while still guaranteeing a fully functional layout that meets all performance specifications. In our approach, the layout tools are driven *directly* by the performance constraints (see Fig. 1(b)). This has several advantages. First, by directly taking into account the high-level performance specifications, a complete and sensible trade-off between the different alternative solutions can be made. Second, since the performance degradation is calculated at run-time, it is not only possible to keep all performance characteristics within their limits, but also to optimize the layout with respect to a subset of the specifications. Finally, while the methodology described in [3] and [4] can lead to a number of iterations with different constraint values, our algorithm will either yield a correct layout or will flag the specifications as being impossible to meet, without iterations. This approach therefore eliminates the feedback route between constraint derivation, placement and layout extraction.

In this paper we present an analog placement tool based on this approach. The tool uses a simulated-annealing-based optimization algorithm, with a cost function designed to drive a random start solution to a placement where all performance constraints are satisfied. Our placement tool differs from other analog placement approaches [1], [2], [5] in that it takes into account symmetry constraints, performance degradation due to interconnect parasitics as well as due to device mismatches,

Manuscript received December 20, 1994; revised April 11, 1994. This work was supported in part by the ESPRIT Project ADMIRE and a Contract with ESA-ESTEC.

The authors are with the Katholieke Universiteit Leuven, Department of Elektrotechniek, ESAT-MICAS, Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium.

IEEE Log Number 9412408.

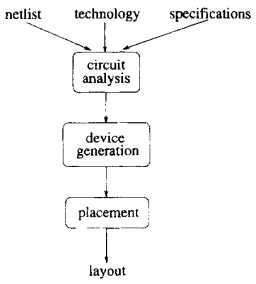


Fig. 2. Program flow of the analog placement tool.

and combines this with the aggressive geometrical optimization techniques (devices merges, abutment, etc.) introduced in [1]. In addition, a number of features have been added to make the tool suitable for use with submicron technology processes in a mixed-signal context.

This paper is organized as follows: Section II describes the general flow of the analog placement tool. Circuit analyses that have to be performed first are discussed in Section III. Section IV describes the device generators that are used in the placement program. The placement algorithm itself is then discussed in Section V and details of how the performance degradation due to parasitics and mismatches are estimated, are presented in Section VI. Examples of practical placements produced with the placement tool are then presented in Section VII. Finally, conclusions of our research are formulated in Section VIII.

II. Overview

The basic components of the placement program are illustrated in Fig. 2. The input to the tool is the circuit netlist after sizing as well as the list of performance specifications that the circuit has to meet (e.g., phase margin $\geq 60°$). The difference between these specified values and the actual performance values obtained by the circuit after sizing are the margins that can be taken up by layout-induced performance degradation. The latter is evaluated using the numerically calculated sensitivities of the considered performances toward the considered layout-dependent performance degrading effects

$$\Delta P_j = \sum_i S^j_{x_i} x_i \tag{1}$$

where ΔP_j is the degradation of the jth performance, $S^j_{x_i} = (\delta P_j)/(\delta x_i)$ the sensitivity of performance P_j to layout effect x_i, evaluated only once in the parasitic-free design point. This linear approximation is valid for small values of the parasitic effect. The placement tool is driven such that the total performance degradations ΔP_j for all performances are below their maximum margins as given by the input specifications

$$\Delta P_j \leq \Delta P_{j,\max} \quad j = 1 \cdots k \tag{2}$$

where k is the number of specifications.

This performance-driven principle can be applied to any layout-dependent performance degrading effect x_i for which the relation between the actual layout and the impact on the performance can be modeled and simulated. In this paper, this will be illustrated for parasitic wire capacitances and resistances, as well as for device mismatches. Note also that although this is only a placement program, the effect of the wiring introduced in the later routing phase is already estimated during the placement, and routing will therefore introduce only an incremental additional degradation of the performance to the extent of the difference between the actual and the estimated wire parasitics.

The first step in the execution of the placement program consists of a number of numerical simulations which result in the set of required performance sensitivities and operating-point information (branch currents, node voltages) of the circuit. This information, together with the circuit netlist, is then used as input for a set of device generators that construct a list of geometrical variants for every device (see Section IV). Only the information needed for the optimization of the placement is generated.

Next, a simulated-annealing algorithm is used to create a placement which respects all circuit specifications. The specific analog features are implemented in the following way:

1) *Symmetry:* Symmetry is considered to be an absolute constraint. If the user specifies a number of devices as being symmetric and/or selfsymmetric, the devices have to be symmetric in the resulting placement. Consequently, symmetry constraints are handled in the move set. Groups of symmetric devices are moved simultaneously such that their symmetry is preserved at all times during the optimization and also in the final result.
2) *Matching:* The impact of mismatches is included in the cost function. The user can specify a pair of devices as being matched without specifying the degree of matching. Matching devices are always generated identically and with identical orientations but it is up to the placement tool to determine the positions and therefore also the distance between the matched devices such that the circuit performance constraints are met. Since it is not always possible in an analog circuit layout to, at the same time, meet all symmetry requirements, put all matching devices directly next to each other and obtain a fairly compact layout, the matching degree of a pair of devices has to be selected in view of its influence on the performance of the circuit. Mismatch is therefore included in the set of parasitic effects for which the degradation of the performance characteristics is calculated and included in the placement cost function.
3) *Circuit Loading:* The performance degradation due to parasitic wire and node capacitance and resistance is also a factor which is included in the placement cost function.
4) *Device Merging:* When it is possible to merge two device terminals, the total junction capacitance of the net to which the terminals belong decreases with an amount proportional to the area and the perimeter of the overlap region. This decreases the total performance degradation and lowers the cost function. In this way, device geometry sharing is promoted specifically for sensitive nets.

After the optimization, the device generators are called again to create the actual layout for the selected geometrical device variants and the final layout is constructed. The output of the program is then the final layout, together with information about the performance degradation in this final layout and an identification of the most important contributions to this degradation. In case the degradation exceeds the required performance specifications, this information can be used by the designer to see the failing performance(s) and to identify the critical effects. This allows him to improve his design when desired.

The relevant details of the different steps will now be discussed in the following sections.

III. Circuit Analysis

An operating-point analysis is first performed on the sized circuit to extract all branch currents and node voltages. The current through a branch is used to determine the necessary width of the routing wires. For small currents, minimal width wires can be used. For larger currents, the wire width has to be adjusted in order to avoid electromigration. This information is used to make accurate estimations for the values of parasitic node capacitances and resistances and to estimate the area which has to be reserved for routing.

Another set of simulations is needed to extract the sensitivities of the circuit specifications with respect to device mismatches and parasitic node capacitances and resistances. The concept of sensitivities and how they are used to calculate performance degradation will be explained in detail in Section VI.

IV. Device Generation

With respect to device generation, we follow the approach outlined in [1]: we only provide device generators for basic circuit devices (transistors, resistors and capacitors) and we rely as much as possible on dynamic geometry sharing techniques to obtain merged structures for multi-device subcircuits (differential pairs, current mirrors, etc.). This drastically reduces the number of generators that needs to be developed and maintained. These device generators can also be used interactively during manual layout.

Before the execution of the optimization algorithm, the device generators are called in interface mode to generate a list of possible geometrical variants for each device, based on the actual device parameters, the technology process and a number of user-specified options. For each variant, the geometrical and electrical information relevant for the execution of the placement algorithm is generated: the bounding box, protection frames for each terminal and the values for parasitic terminal capacitances. The terminal protection frames are used to detect device merges as described in [1]. The parasitic terminal capacitances are needed to calculate performance degradation due to interconnect parasitics (see Section VI). They can be calculated accurately since the dc voltages of the connecting nodes are known.

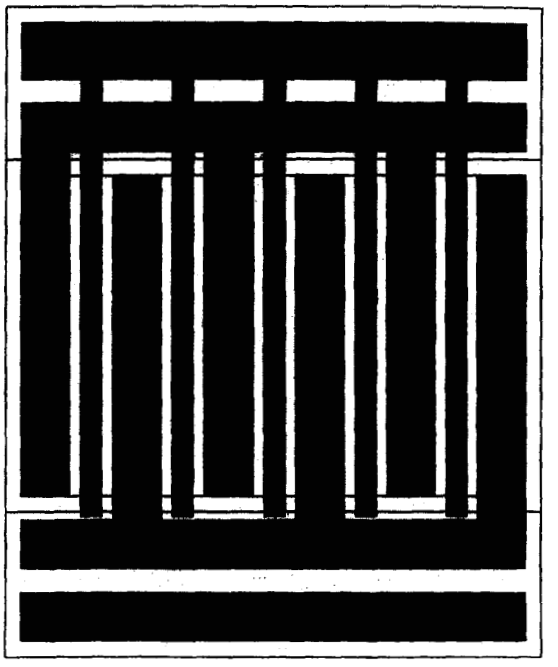


Fig. 3. A regular finger-style layout for a MOS transistor.

The protection box of a device is calculated by taking the physical bounding box of the device and expanding it on each side by a certain amount to reserve space for routing. The necessary routing space is estimated for each side based on the width of the wires which must be connected to the terminal on that side. By taking into account variable wire widths, we are able to make fairly accurate routing space estimates.

We have added special features to the device generators in order to make them suitable for use in a mixed-signal context. Switching transients in digital MOS circuits can perturb analog circuits integrated on the same die by means of coupling through the substrate. An effective way to protect sensitive devices against coupling noise is to make sure that no part of a MOS transistor is more than a specified minimum distance away from a bulk contact. This minimum distance can be a design rule imposed by the foundry or can be specified by the designer. Our MOS transistor layout generator can handle this kind of constraint. When necessary, the MOS transistor will be split in a number of parallel transistors and bulk straps will be added between the different parts. Fig. 3 shows a regular MOS finger style transistor, while Fig. 4 shows a large transistor which has been split into parallel parts. Also, the wire width of the source and drain wires of the last transistor has been automatically adapted to the large current.

After the execution of the placement algorithm, the device generator is called in layout mode for the selected variant. This time, the generator returns the actual layout of the selected variant of the device together with a model that includes all second-order parasitic elements. This model can then be backannotated to the netlist to carry out more accurate simulations.

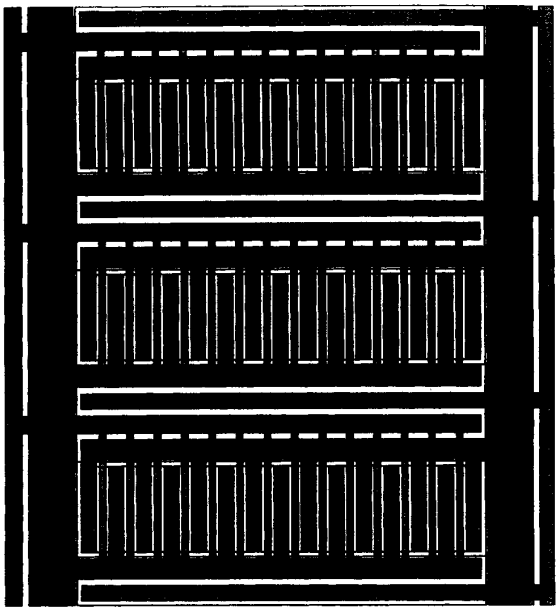


Fig. 4. A special layout structure for a large MOS transistor. The transistor is split into parallel parts and width of the source and drain wires has been adapted to large currents.

V. The Analog Placement Algorithm

A. The Simulated Annealing Algorithm

The optimization algorithm used for the performance-driven placement program is simulated annealing [6], a general and robust optimization technique that uses stochastically controlled hill-climbing to avoid the many local minima in complex cost surfaces and thus to find better global solutions. The algorithm works by generating a large number of randomly selected perturbations on an initial random placement. For each proposed move, the effect that the move has on the cost of the circuit is evaluated. If the proposed move decreases the cost function, it is accepted. If, however, the proposed move increases the cost function C, it is accepted with a probability that is affected by the temperature T, a control parameter that is decreased gradually in the course of the annealing process, in the following manner:

$$P[uphill] \sim e^{(-\Delta C)/T}. \tag{3}$$

The important aspects of the annealing algorithm are discussed in the following subsections.

B. Placement Representation

We use a flat, nonslicing placement representation with the absolute coordinates, the orientation and the geometrical variant of each device (or device cluster) as annealing variables. For analog performance-driven placement this model is to be preferred over the alternative slicing-style representation for several reasons. The main reasons are that an accurate estimation of parasitic circuit loading and mismatch effects requires the knowledge of absolute device coordinates, which are not known in a slicing structure, and secondly that a slicing-style placement model limits the set of reachable device configurations, prohibits geometry-sharing optimizations and makes it difficult to implement symmetry constraints.

C. Move Set

The following moves are executed during placement optimization:

1) *Device Translation:* A device is chosen at random and its center position is translated to a randomly selected coordinate.
2) *Device Reorientation:* A device is chosen at random and its orientation is changed to one of the eight orthogonal rotation and/or mirror values possible by combining 90° rotations with mirroring in the X and Y direction.
3) *Device Swap:* Two devices are chosen at random and their center coordinates are interchanged.
4) *Device Reshape:* A device is chosen at random and casted into a new geometrical variant, chosen from the list created by the device generator.

In addition to this classic single-device move set, a number of moves which operate on clusters of matched, merged or symmetric devices have also been implemented. These moves simultaneously change the position, orientation or the variant of a set of clustered devices.

D. Cost Function

The search for an optimum placement is driven by the cost function. This cost function C is calculated for each intermediate placement result and is a weighted sum of 4 terms

$$C = \alpha C_{area} + \beta C_{aspect\ Ratio} + \gamma C_{overlap} + \delta C_{perf\ Degr} \tag{4}$$

where

1) C_{area}: the area of the bounding box of the intermediate placement. This term is used to minimize the area.
2) $C_{aspectRatio}$: this term is proportional to the deviation of the aspect ratio of the intermediate placement to the aspect ratio specified by the user.
3) $C_{overlap}$: this term is proportional to the total amount of illegal overlap present in the intermediate placement and is driven to zero in the final placement.
4) $C_{perfDegr}$: this term is used to keep the performance degradation induced by interconnect parasitics and device mismatches within user-specified limits. This term is extremely important for our placement approach and is therefore described in detail in the next section.

The weighting coefficients α, β, γ, and δ are adapted dynamically during the placement optimization.

VI. Estimating Performance Degradation

We now describe how the performance degradation of an analog circuit due to interconnect parasitics and device mismatches, is estimated in the program. Note, however that the same technique can be applied to other layout-dependent

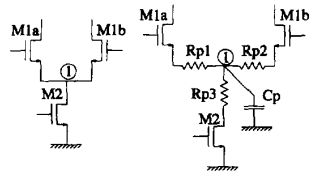


Fig. 5. Modeling of interconnect parasitics on node 1. (a) Differential pair without parasitics. (b) Differential pair with parasitic capacitor and resistors.

performance-degrading effects, provided that the effect can be quantified.

In each case we follow the same three-step methodology. First we extract the relevant geometrical information out of the intermediate placement solution. Based on this information we calculate the value of the parasitic circuit elements which model the parasitic effect during simulation. Finally, we calculate the influence of the parasitic circuit elements on the performance characteristics of the circuit using a linear approximation (valid for small values of the parasitic effect) based on performance sensitivities, which have been calculated in advance before the start of the placement optimization. This methodology will now be applied to interconnect parasitics and device mismatches.

A. Interconnect Parasitics

An n-terminal net can be modeled by a starred configuration of n parasitic resistors and 1 parasitic capacitor (see Fig. 5). The values of the capacitor and the resistors can only be approximated since the actual geometry of the wire is unknown before routing. A minimum spanning tree connecting all terminals is calculated and the sum of the lengths of its paths is used as an approximation for the total net length L.

The parasitic capacitance C_p is the sum of two components

$$C_p = C_j + C_w. \tag{5}$$

C_j is caused by the sum of the junction capacitances of all connecting terminals and C_w is caused by the capacitance to the substrate of the actual wire.

The parasitic resistances $R_{p,i}, i = 1, \cdots, n$ of the wire segments are calculated as follows:

$$R_{pi} = \rho_{square} \frac{\frac{L}{n}}{W_{wire,i}} \tag{6}$$

where ρ_{square} is the sheet resistance of the primary routing layer, L is the estimated total net length, and n the number of terminals connected to the net. $W_{wire,i}$ is the width of the ith wire segment which can be calculated from the current flowing through the ith terminal.

Once the value of C_p and $R_{p,i}, i = 1, \cdots, n$ is known for every net, the performance degradation ΔP_j for the jth performance characteristic due to interconnect parasitics can be determined using the precalculated sensitivity information

$$\Delta P_j = \sum_{k=1}^{m} \left(S^j_{C_{p,k}} C_{p,k} + \sum_{i=1}^{n_k} S^j_{R_{p,ki}} R_{p,ki} \right) \tag{7}$$

where m is the number of nodes minus the ground node and n_k is the number of terminals of net k. $S^j_{C_{p,k}} = [(\delta P_j)/(\delta C_{p,k})]$ and $S^j_{R_{p,ki}} = [(\delta P_j)/(\delta R_{p,ki})]$ are the sensitivities of performance characteristic P_j to small changes in the parasitic capacitance $C_{p,k}$ and the parasitic resistances $R_{p,ki}$. These sensitivities are determined in advance by simulation.

B. Mismatches

We follow a comparable approach with respect to mismatches. $\sigma^2(V_{T0})$ and $\sigma^2(\beta)$ for the V_{T0} and β mismatch of MOS transistors can be estimated using the following equations [7]:

$$\sigma^2(V_{T0}) = \frac{A^2_{V_{T0}}}{WL} + S^2_{V_{T0}} D^2 \tag{8}$$

$$\sigma^2(\beta) = \frac{A^2_{\beta}}{WL} + S^2_{\beta} D^2 \tag{9}$$

where $A_{V_{T0}}$, $S_{V_{T0}}$, A_β, and S_β are constants depending on the process. The area of the devices WL and the distance D are known for each intermediate placement. Based on this information, the standard deviations of V_{T0} and β can be calculated with (8) and (9).

Using sensitivity information, the effect on the performance degradation can be estimated as follows:

$$\Delta P_j = \sum_{k=1}^{m} \{ S^j_{\Delta V_{T0,k}} [3\sigma(V_{TO})_k] + S^j_{\Delta\beta_k} [3\sigma(\beta)_k] \} \tag{10}$$

where $S^j_{\Delta V_{T0,k}} = [(\delta P_j)/ (\delta \Delta V_{T0,k})]$ and $S^j_{\Delta\beta_k} = [(\delta P_j)/ (\delta \Delta \beta_k)]$ are the sensitivities of performance characteristic P_j to small changes in ΔV_{T0} and $\Delta\beta$ of matching transistor pair k. The sum is taken over all pairs of matching devices.

Equation (10) can be rewritten as

$$\Delta P_j = \Delta P_{j,area} + \Delta P_{j,distance}. \tag{11}$$

$\Delta P_{j,area}$ represents the degradation of the jth performance characteristic due to area effects. This term can be computed after sizing and remains constant during placement.

$\Delta P_{j,distance}$ represents the degradation of the jth performance characteristic due to distance effects and therefore depends on the actual layout. This term can be computed as follows:

$$\Delta P_{j,distance} = \sum_{k=1}^{m} [S^j_{D_k} (D_k)] \tag{12}$$

where D_k represents the distance between the transistors of matching pair k and $S^j_{D_k}$ is the sensitivity of performance characteristic j to small variations in distance D_k. This term must be recomputed for every new placement.

VII. Experimental Results

The algorithm described in this paper has been implemented using the C++ language in the UNIX environment and has been integrated in a complete synthesis environment for analog

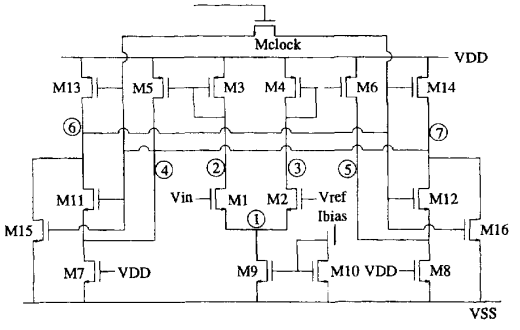


Fig. 6. High speed CMOS comparator.

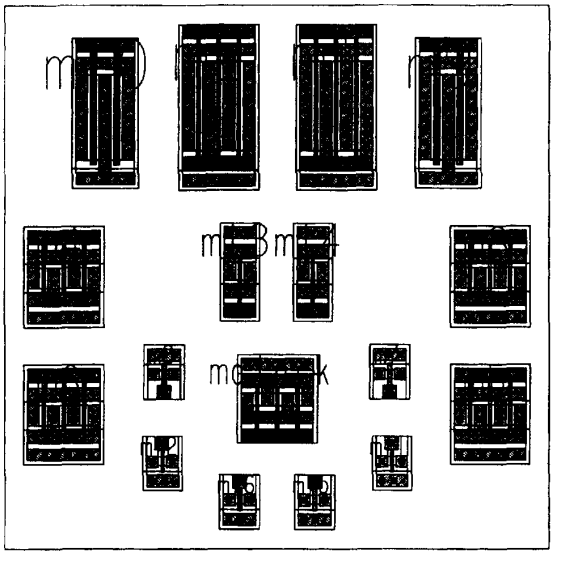


Fig. 7. CMOS comparator: placement 1: generated automatically by the performance-driven placement program.

circuits [11]. The program was tested on a number of practical circuits. Three of them are presented in this section.

The first example is a high-speed CMOS comparator [8]. The circuit is used in a CMOS A/D converter and its performance is a limiting factor for the performance of the overall A/D converter. The specifications imposed upon the circuit are a propagation delay of less than 5 ns and an offset voltage of less than 5 mV. The circuit schematic is shown in Fig. 6. To demonstrate the effectiveness of our direct performance-driven approach we have generated two placements for this circuit. Placement 1 (see Fig. 7) was generated with the presented performance-driven placement tool, while placement 2 was generated in the traditional way, with the same placement tool but with the performance-driven mechanism disabled. It can be seen from Table I that the simulated performance of placement 1 is significantly better than that of placement 2. Placement 1 has both performance characteristics within the user-specified ranges, while for placement 2 both specifications are violated. The optimized distances between the matching transistor pairs together with the resulting offset voltage degradation due to distance effects are shown in Table II for

TABLE I
COMPARISON OF THE PERFORMANCE OF TWO AUTOMATICALLY GENERATED COMPARATOR PLACEMENTS

performance				
Performance	Spec	Plac 1	Plac 2	Unit
offsetvoltage	< 5	3.7	6.9	*mV*
delay	< 5	2.8	5.4	*nsec*

TABLE II
PLACEMENT 1 (PERFORMANCE DRIVEN): OFFSET VOLTAGE DEGRADATION DUE TO DISTANCE EFFECTS

comparator offset voltage degradation			
Transistor Pair	Distance	Sensitivity	Degradation
	μm	$\frac{\mu V}{\mu m}$	mV
M1 – M2	60	12	.720
M3 – M5	70	2.9	.203
M4 – M6	70	2.9	.203
M7 – M8	118	.2	.024
M11 – M12	119	1.93	.230
M13 – M14	38	3.8	.145
M15 – M16	40	3	.120
total			1.645

TABLE III
PERFORMANCE CHARACTERISTICS OF THE FULLY DIFFERENTIAL CMOS OPAMP

opamp performance				
Performance	Specification	Nominal Value	After Placement	Unit
A_v	> 100	107	104	*dB*
GBW	> 200	205	202	*MHz*
PM	> 70	77	74	deg
CMRR@10Hz	> 70	∞	78	*dB*
PSRR@10Hz	> 80	∞	86	*dB*

placement 1. The nominal values are the values obtained after sizing of the circuit without parasitic layout effects (no parasitic node capacitances and no mismatch). It can be seen that the performance-driven algorithm selectively minimizes the distances for the most sensitive transistor pairs, which results in a lower offset voltage. CPU times were 106 and 93 s for placement 1 and 2, respectively, on a SUN SPARC 10 workstation, which means that the performance-driven mechanism significantly improves the circuit performance at only a small increase in CPU time.

As a second example, a fully differential CMOS operational amplifier [9] (see Fig. 8) was used to test the efficiency of the algorithm for larger circuits. The placement of the opamp is shown in Fig. 9. Note the clear symmetry axis in this fully differential circuit. The circuit's specifications together with the obtained performance after placement are given in Table III. The degradation of all performances clearly remains within the specified margins. This placement required a CPU time of 163 s (less than 3 min) on a SUN SPARC 10 workstation.

The third example is a high-speed CMOS operational amplifier [10]. The schematic of the circuit is shown is Fig. 10. The placement that was generated for this circuit is shown in Fig. 11. The symmetry axis of the circuit is clearly visible in the placement. This circuit contains some very large transistors which have been split into parallel parts for reasons

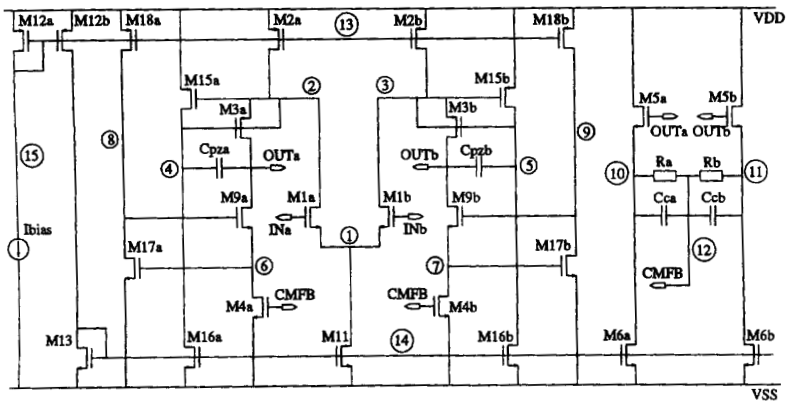


Fig. 8. CMOS fully differential opamp.

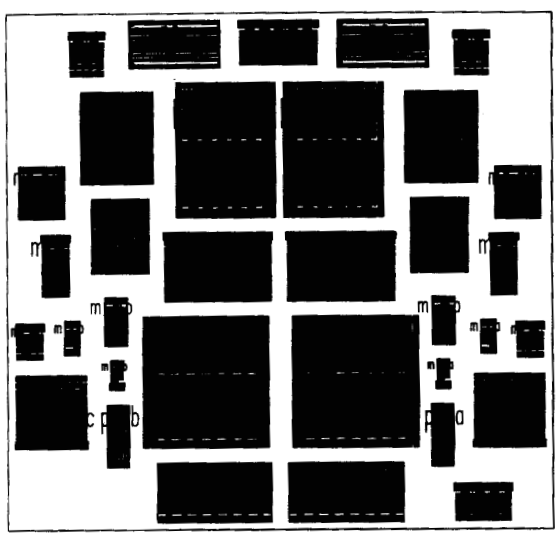


Fig. 9. Placement of the CMOS operational amplifier generated automatically by the performance-driven placement program.

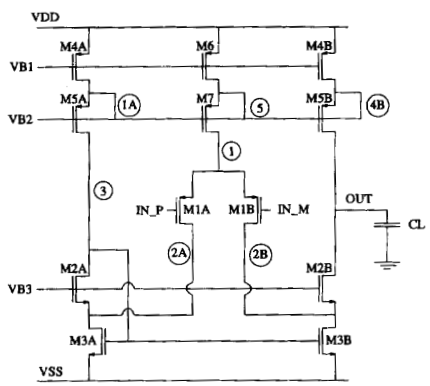


Fig. 10. CMOS high-speed opamp.

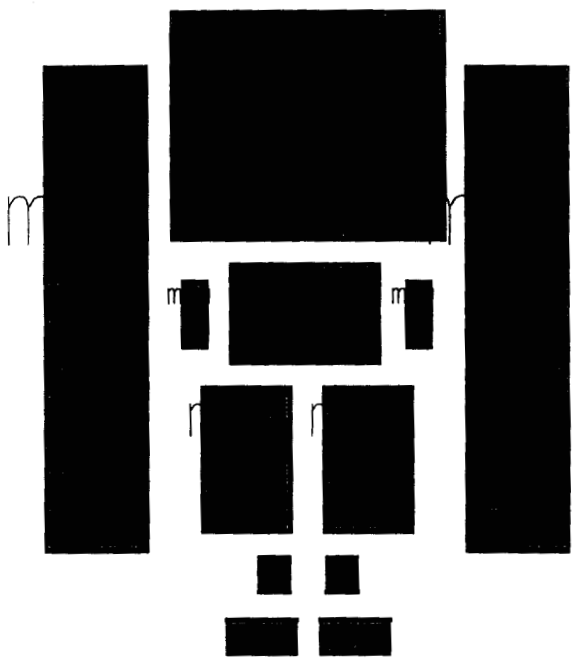


Fig. 11. Placement of the high-speed CMOS operational amplifier generated automatically by the performance-driven placement program.

TABLE IV
PERFORMANCE CHARACTERISTICS OF THE HIGH-SPEED CMOS OPAMP

opamp performance				
Performance	Specification	Nominal Value	After Placement	Unit
GBW	> 225	228	226	*MHz*
A_v	> 60	67	67	*dB*
PM	> 60	61	60.2	deg
slew − rate	> 150	163	163	$V/\mu sec$
Voffset	< 5	4.5	4.8	*mV*

explained in Section IV. As shown in Table IV all performance characteristics for this circuit are within the specifications. All specifications were met in one pass of the program (CPU time 83 s).

VIII. CONCLUSIONS AND FUTURE WORK

A new direct performance-driven placement tool for analog integrated circuits has been presented. A simulated annealing

algorithm is used to optimize a placement while keeping the layout-induced performance degradation within the margins imposed by the designer's specifications. The cost assigned to an intermediate placement is based on an accurate estimation of the performance degradation caused by interconnect parasitics and by device mismatches. This technique can easily be extended to other effects as well. Experimental results have demonstrated the feasibility and efficiency of our approach.

Future work includes the extension of the methodology to analog routing and the integration of performance-driven placement and routing for analog circuits, to avoid possible layout iterations due to the additional performance degradation caused by the actual wiring.

REFERENCES

[1] J. M. Cohn, R. A. Rutenbar, and L. R. Carley, "KOAN/ANAGRAM II: New tools for device-level analog placement and routing," *IEEE J. Solid-State Circuits*, vol 26, no. 3, pp. 330–342, Mar. 1991.

[2] V. K. Meyer zu Bexten, C. Moraga, and R. Klinke, "ALSYN: Flexible rule-based layout synthesis for analog IC's," *IEEE J. Solid-State Circuits*, vol. 28, no. 3, pp. 261–268, Mar. 1993.

[3] 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, no. 2, pp. 208–224, Feb. 1993.

[4] E. Charbon, E. Malavasi, and A. Sangiovanni-Vincentelli, "Generalized constraint generation for analog circuit design," in *Proc. IEEE ICCAD*, Nov. 1993, pp. 408–414.

[5] E. Charbon, E. Malavasi, U. Choudhury, A. Casotto, and A. Sangiovanni-Vincentelli, "A constraint-driven placement methodology for analog integrated circuits," in *Proc. IEEE CICC*, May 1992, pp. 28.2.1–28.2.4.

[6] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.

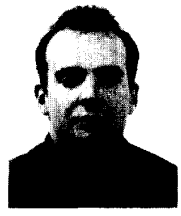
[7] M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers, "Matching properties of MOS transistors," *IEEE J. Solid-State Circuits*, vol. 24, no. 5, pp. 1433–1440, Oct. 1989.

[8] M. Steyaert, R. Roovers, and J. Craninckx, "A 110 MHz 8 bit CMOS interpolating A/D converter," in *Proc. IEEE Custom Integrated Circuits Conf.*, May 1993, pp. 28.1.1–28.1.4.

[9] E. Peeters and K. Ghafoor, "Design of a fully differential high-speed CMOS amplifier," Master's thesis, Katholieke Universiteit Leuven, Dept. Elektrotechniek, Heverlee, Belgium, June 1993.

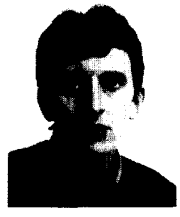
[10] J. Fisher and R. Koch, "A highly linear CMOS buffer amplifier," *IEEE J. Solid-State Circuits*, vol. SC-22, no. 3, pp. 330–334, June 1987.

[11] G. Gielen *et al.*, "An analog module generator for mixed analog/digital ASIC design," to be published in *Int. J. Circuit Theory Applicat.*, Special Issue on Analog Tools for Circuit Design, 1995.



Koen Lampaert was born in Roeselare, Belgium, in 1968. He received the M.Sc. degree in electrical and mechanical engineering in 1992 from the Katholieke Universiteit Leuven, Belgium.

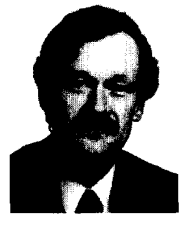
He is currently with the ESAT-MICAS laboratories of the Katholieke Universiteit Leuven as a Research Assistant. He is working towards the Ph.D. degree on automated analog layout. His research interests are in analog design automation.



Georges Gielen (M'92) received the M.Sc. and Ph.D. degrees in electrical engineering from the Katholieke Universiteit Leuven, Belgium, in 1986 and 1990, respectively.

In 1990, he was with the Department of EECS of the University of California at Berkeley as a Postdoctoral Research Assistant and Visiting Lecturer. From 1991 to 1993, he was a Postdoctoral Research Assistant of the Belgian National Fund of Scientific Research at the ESAT laboratory of the Katholieke Universiteit Leuven. In 1993, he was with the Katholieke Universiteit Leuven as a Tenure Research Associate of the Belgian National Fund of Scientific Research and as an Assistant Professor. His current research interests are in the design of analog and mixed-signal integrated circuits, and especially in analog and mixed-signal CAD (numerical and symbolic simulation, synthesis, layout, and design for manufacturability) and test. He is a Technical Coordinator of several industrial research projects in this area.

Dr. Gielen serves regularly on the Program Committee of international conferences and he is currently an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I: FUNDEMENTAL THEORY AND APPLICATIONS. He has authored or coauthored one book and more than 50 papers in edited books, international journals, and conference proceedings.



Willy M. Sansen (S'66–M'72–SM'86) received the M.S. degree in electrical engineering from the Katholieke Universiteit Leuven in 1967 and the Ph.D. degree in electronics from the University of California at Berkeley in 1972.

Since 1981 he has been with the ESAT laboratory of KU Leuven as a Full Professor. Between 1984–1990, he was the Head of the Electrical Engineering Department. He was a Visiting Professor with Stanford University, Stanford, CA, in 1978, at the Federal Technical University Lausanne in 1983, at the University of Pennsylvania, Philadelphia in 1985, and at the Technical University Ulm in 1994. He has been involved in design automation and in numerous analogue integrated circuit designs for telecom, consumer electronics, medical applications, and sensors. He has been supervisor of 30 Ph.D. theses in that field.

Dr. Sansen has authored and coauthored more than 300 papers in international journals and conference proceedings and six books. He has also coauthored (with K. Laker) a textbook, *Design of Analog Integrated Circuits and Systems*.