Direct Performance-Driven Placement of Mismatch-sensitive Analog Circuits

K. Lampaert, G. Gielen*, W. Sansen

Katholieke Universiteit Leuven, Dep. Elektrotechniek, ESAT-MICAS

Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium

Tel.: +32/16/220931 - Telefax: - +32/16/221855

*research associate of the National Fund of Scientific Research

Abstract

This paper presents a direct performance-driven placement algorithm for analog integrated circuits. The performance specifications directly drive the layout tools without intermediate parasitic constraints. A simulated-annealing algorithm is used to drive an initial solution to a placement that respects the circuit's performance specifications. During each iteration, the layout-induced performance degradation is calculated from the geometrical properties of the intermediate solution. The placement tool handles symmetry constraints, circuit loading effects and device mismatches. The feasibility of the approach is demonstrated with practical circuit examples.

1. Introduction

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 unavoidable parasitics associated with the interconnections they can introduce intolerable performance degradation. The main concern in automated analog layout synthesis is therefore to control the effects of parasitics on circuit performance and to keep the layout-induced performance degradation within user defined limits, as specified in the circuit's performance specifications.

Many existing layout programs, like KOAN/ANAGRAM[1] or ALSYN[2], however, try to optimize the layout without quantifying the performance degradation. They therefore can not guarantee that the resulting layout will also meet the specifications. In recent years on the other hand, a methodology has been introduced [3, 4] which tries to achieve this by mapping the performance specifications onto a set of constraints on 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 then 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.

32nd ACM/IEEE Design Automation Conference ®

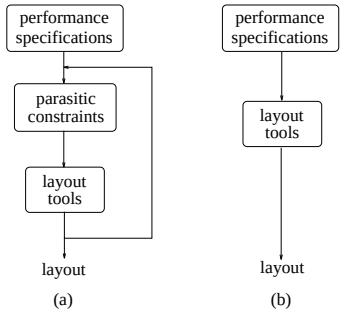

Fig. 1: Performance-driven layout methodologies:
(a) with an intermediate constraint generation step
(b) performance specifications directly taken into account by the layout tools

We 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 [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 too tight, without iterations.

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 similar approaches in that it takes into account symmetry constraints, performance degradation due to interconnect parasitics as well as due to device mismatches and combines this with the aggressive geometrical optimization techniques (devices merges, abutment..) introduced in [1]. In addition, a number of features have been added to make the tool suitable for use with sub-micron processes in a mixed signal context.

This paper is organized as follows. Section 2 describes the general flow of the analog placement tool. Section 3 discusses the device generators that are used in the placement program. The placement algorithm itself is then discussed in section 4 and details of how the per-

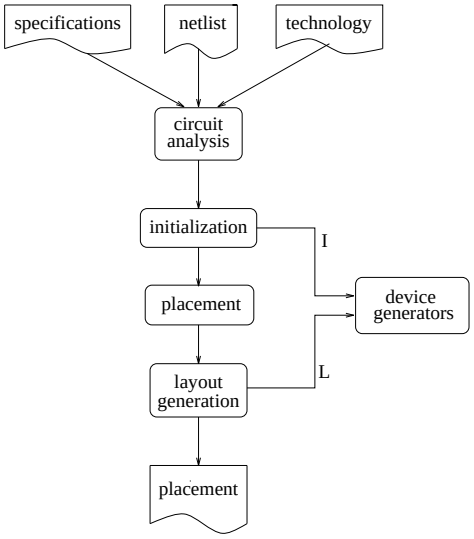

Fig. 2: Program flow of the analog placement tool

formance degradation due to parasitics and mismatches are estimated are presented in section 5. Examples of practical placements produced with the placement tool are then presented in section 6. Finally, conclusions of our research are formulated in section 7.

2. Overview

The basic components of the placement algorithm are illustrated in Fig. 2. The first step in the execution of the algorithm consists of a number of numerical simulations which result in a set of performance sensitivities and operating-point information about the circuit (branch currents, node voltages). This information together with the circuit netlist is then used as input for a set of device generators that construct a list of variants for every device (see section 3). Only the information needed for the optimization of the placement is generated.

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

- Symmetry : We consider symmetry to be an absolute constraint. If the user specifies a number of devices as being symmetric and/or selfsymmetric, that symmetry is preserved at all times during the optimization and also in the final result, by handling it in the move set.
- Matching : In contrast to symmetry, matching is not an absolute constraint. The user can specify a pair of devices as being matched without specifying the degree of matching. It is up to the placement tool to determine the position and orientation of the matched devices such that the circuit performance constraints are met. The matching degree of a pair of devices has to be selected in view of its influence on the performance of the circuit. Therefore we include mismatch in the set of parasitic effects of which the influence on performance characteristics has to be calculated in the cost function.
- Circuit loading : The effect of parasitic node capacitance and resistance is also a factor which is included in the cost function.
- 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, geometry sharing is promoted specifically for sensitive nets.

After the optimization, the device generators are called again to create the actual geometry for the selected variants and the final layout is constructed. The relevant details of the different steps will now be discussed in the next sections.

3. Device Generation

With respect to device generation, we follow the approach outlined in [1] : we only provide device generators for the different variants of 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...). This drastically reduces the number of generators that needs to be developed and maintained.

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. When necessary, the MOS transistor will be split in a number of parallel transistors and bulk straps will be added between the different parts.

4. The Analog Placement Algorithm

The optimization algorithm used for the performance-driven placement program is simulated annealing [5], 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. The important aspects of the annealing algorithm, the placement representation, the move set and the cost function are discussed in the following subsections.

A. Placement Representation

We use a flat, nonslicing placement representation. For analog performance-driven placement this model is to be preferred over the alternative slicing-style representation for several reasons, mainly because an accurate estimation of parasitic circuit loading and mismatch effects requires the knowledge of absolute device coordinates and because a slicing-style placement model limits the set of reachable device configurations, prohibits geometry-sharing optimizations and makes it difficult to implement symmetry constraints.

B. Move Set

The following moves are executed during placement optimization:

- device translation : a device is chosen at random and its position is translated to a randomly selected coordinate.
- 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.
- device swap : two devices are chosen at random and their center coordinates are interchanged.

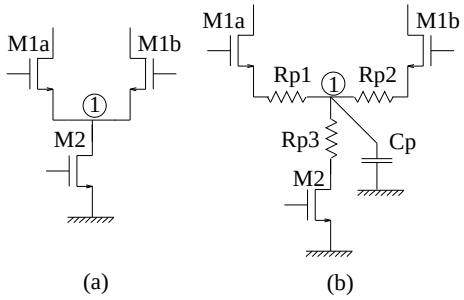

Fig. 3: Modeling of interconnect parasitics on node 1 :
(a) differential pair without parasitics
(b) differential pair with parasitic capacitor and resistors

- device reshape : a device is chosen at random and casted into a new variant, chosen from the list created by the device generator.

In addition to this classical 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 all clustered devices.

C. Cost Function

The search for an optimum placement is driven by the cost function. This cost function C is a weighted sum of 4 terms:

- C_{area} : the area of the bounding box of the intermediate placement. This term is used to minimize the area.
- $C_{aspectRatio}$: this term is proportional to the deviation of the aspect ratio of the intermediate placement to the aspect ratio specified by the user.
- $C_{overlap}$: 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.
- $C_{perfDegr}$: 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.

5. Estimating Performance Degradation

In this section we demonstrate how the performance degradation of an analog circuit due to interconnect parasitics and device mismatches can be estimated. In both cases we follow the same methodology. First we extract the relevant geometrical information out of the intermediate placement solution. Based on this information we calculate the influence of the parasitic effects on the performance characteristics of the circuit using a linear approximation based on performance sensitivities. This methodology will now be applied for 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. 3). 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 netlength L.

The parasitic capacitance C_p is the sum of two components :

$$C_p = C_j + C_w \tag{1}$$

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 $R_{p,i}, i = 1..n$ of the wire segments are calculated as follows :

$$R_{pi} = \rho_{square} \frac{\frac{L}{n}}{W_{wire,i}} \tag{2}$$

where ρ_{square} is the sheet resistance of the primary routing layer, L is the estimated total netlength 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 $R_{p,i}, i = 1..n$ 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:

$$\Delta P_j = \sum_{k=1}^{m} (S^j_{C_{p,k}} C_{p,k} + \sum_{i=1}^{n_k} S^j_{R_{p,ki}} R_{p,ki}) \tag{3}$$

where m is the number of nodes minus the ground node and n_k is the number of terminals of net k. $S^j_{C_{p,k}} = \frac{\delta P_j}{\delta C_{p,k}}$ and $S^j_{R_{p,ki}} = \frac{\delta P_j}{\delta R_{p,ki}}$ 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 [6] :

$$\sigma^2(V_{T0}) = \frac{A^2_{V_{T0}}}{WL} + S^2_{V_{T0}} D^2 \tag{4}$$

$$\sigma^2(\beta) = \frac{A^2_{\beta}}{WL} + S^2_{\beta} D^2 \tag{5}$$

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 (4) and (5).

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

$$\Delta P_j = \sum_{k=1}^{m} (S^j_{\Delta V_{T0,k}} (3\sigma(V_{TO})_k) + S^j_{\Delta\beta_k} (3\sigma(\beta)_k) \tag{6}$$

where $S^j_{\Delta V_{T0,k}} = \frac{\delta P_j}{\delta \Delta V_{T0,k}}$ and $S^j_{\Delta\beta_k} = \frac{\delta P_j}{\delta \Delta\beta_k}$ 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 6 can be rewritten as

$$\Delta P_j = \Delta P_{j,area} + \Delta P_{j,distance} \tag{7}$$

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

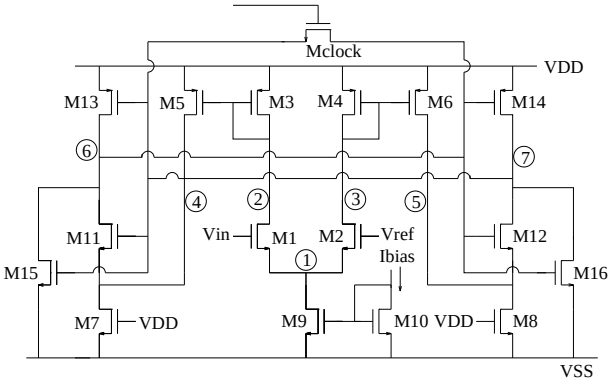

Fig. 4: Comparator

$\Delta P_{j,distance}$ 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 :

$$\Delta P_{j,distance} = \sum_{k=1}^{m} (S^j_{D_k}(D_k)) \qquad (8)$$

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.

6. Experimental results

The algorithm described in this paper has been implemented using the C^{++} language in the UNIX environment and tested on a number of practical circuits. Two of them are presented in this section.

The first one is a high-speed CMOS comparator [7]. 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 $5nsec$ and an offset voltage of less than $5mV$. The circuit schematic is shown in Fig. 4. To demonstrate the effectiveness of our direct performance-driven approach we have generated two placements for this circuit. Placement 1 (see Fig. 5) was generated with the performance-driven algorithm, while placement 2 (see Fig. 6) was generated with the performance-driven mechanism disabled. It can be seen from Table 1 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 one specification is violated. The optimized distances for the matching transistor pairs together with the resulting offset voltage degradation due to distance effects are shown in Table 2 for placement 1 and in Table 3 for placement 2. 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 seconds, respectively on a SUN SPARC 10 workstation, which means that the performance-driven mechanism largely improves the circuit performance at only a small increase in CPU time.

As a second example, a fully differential CMOS operational amplifier [8] (see Fig. 7) was used to test the efficiency of the algorithm for larger circuits. The placement of the opamp is shown in Fig. 8. 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 4. The layout-induced

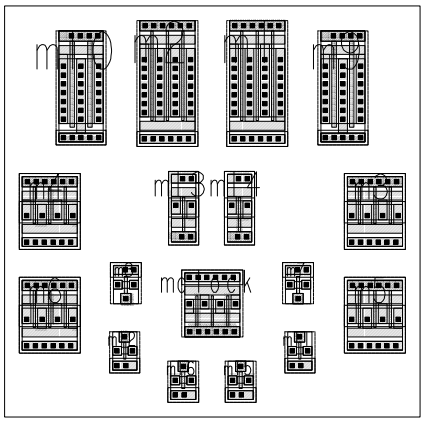

Fig. 5: CMOS comparator : placement 1 : performance driven

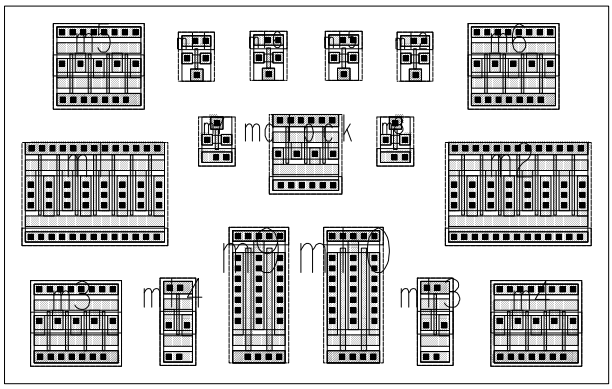

Fig. 6: CMOS comparator : placement 2 : not performance driven

parasitics degrade the phase margin with 3 degrees, but the performance driven mechanism makes that this is well below the maximum allowed 7 degrees of degradation. The actual routing (i.e. the difference with the estimated wire parasitics) can still further degrade this phase margin. This placement required a CPU time of 163 seconds on a SUN SPARC 10 workstation.

7. Conclusions and Future Work

A new direct performance-driven analog placement tool has been presented. A simulated annealing algorithm is used to optimize a placement with respect to performance. 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. The performance degradation is computed at run-time, which provides maximum flexibility for the placement tool. The method combines a reasonable accuracy with the efficiency needed for use in a simulated-annealing loop. Experimental results demon-

performance				
Performance	Spec	Plac 1	Plac 2	Unit
offset voltage	< 5	3.7	6.9	mV
delay	< 5	2.8	5.4	$nsec$

Table 1: Comparison of the performance of two automatically generated comparator placements.
Placement 1 : performance driven
Placement 2 : not performance driven

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 2: 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
$M1-M2$	250	12	3.0
$M3-M5$	151	2.9	.438
$M4-M6$	151	2.9	.438
$M7-M8$	103	.2	.021
$M11-M12$	126	1.93	.243
$M13-M14$	147	3.8	.559
$M15-M16$	43	3	.129
total			4.828

Table 3: placement 2 (not performance driven) : offset voltage voltage degradation due to distance effects.

strate the feasibility and efficiency of our approach.

Future work includes the extension of the methodology to analog routing.

Acknowledgments

Parts of this research are supported by ESA-ESTEC and by the CEC ESPRIT project ADMIRE.

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, pp. 330-342, no. 3, March 1991.

[2] V. K. zu Bexten, C. Moraga, R. Klinke, "ALSYN: flexible rule-based layout synthesis for analog IC's," IEEE J. of Solid State Circuits, vol. 28, no. 3, March 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, February 1993.

[4] E. Charbon, E. Malavasi and A. Sangiovanni-Vincentelli, "Generalized Constraint Generation for Analog Circuit Design," proc. IEEE ICCAD, pp. 408-414, November 1993.

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

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

[7] M. Steyaert, R. Roovers and J. Craninckx, "A 110 MHz 8 bit CMOS Interpolating A/D converter," Proc IEEE Custom Integrated Circ. Conf., pp. 28.1.1-28.1.4 May 1993.

[8] E. Peeters, Ghafoor K., "Design of a Fully Differential High-Speed CMOS Amplifier," KU Leuven Masters Thesis. June 1993.

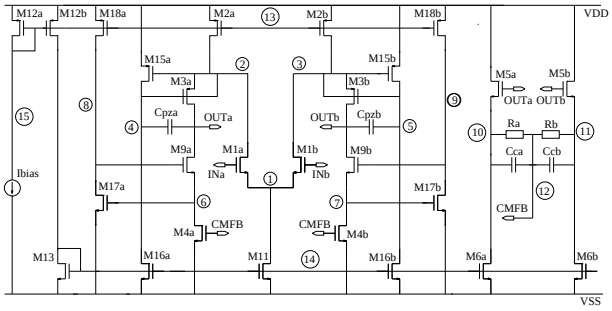

Fig. 7: CMOS differential opamp

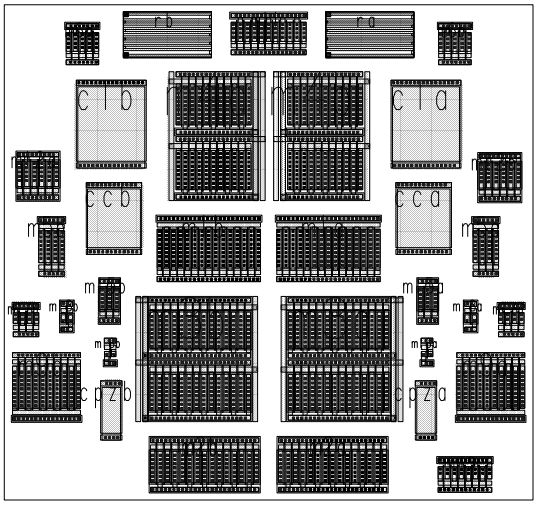

Fig. 8: Opamp placement

opamp performance				
Performance	Spec	Nominal	After Plac	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

Table 4: Performance characteristics of the opamp.