Simultaneous Placement and Module Optimization of Analog IC's

Edoardo Charbon, Enrico Malavasi[†], Davide Pandini[†] and Alberto Sangiovanni-Vincentelli
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720, USA
[†]Dipartimento di Elettronica e Informatica
Università di Padova, 35131 Padova, Italy

ABSTRACT

New placement techniques are presented which substantially improve the process of automatic layout generation of analog IC's. Extremely tight specifications can be enforced on high-performance analog circuits by using simultaneous placement and module optimization. An algorithmic approach to module generation provides alternative sets of modules optimized with respect to area and performance but equivalent in terms of parasitics and topology. The final module selection is performed during the placement phase, based on Simulated Annealing. The flexibility of the annealing algorithm has been significantly improved, thus making it possible to more efficiently exploit the tradeoffs between area, parasitics and matching.

1 Introduction

Layout design automation of analog IC's has seen considerable improvements in recent years despite a continuous increase of complexity and sophistication of analog and mixed-signal systems. On the one hand, denser and more advanced technologies have led to faster and more reliable circuits. On the other hand, important challenges have arisen due to significantly higher performance sensitivity to the details of the physical layout. Digitally targeted layout tools are often ineffective in controlling performance due to the lack of parasitic minimization. Techniques derived from the digital world addressing analog-specific constraints, have been proposed in [1, 2, 3]. These tools focused mainly on the minimization of area and wiring, with no explicit provision to control performance. As a result a large number of time-consuming layout-extraction-simulation iterations may be necessary to meet a set of tight performance specifications.

Only recently performance-driven approaches to the layout of analog circuits have been presented. In [4] and [5] a methodology for the generation and the use of a complete set of analog-specific constraints for the layout synthesis of analog circuits was proposed. The main disadvantage of this implementation however, is the use of the traditional partition of layout synthesis into its basic phases, namely module generation, placement, routing and compaction. Due to the process inherent sequentiality, each phase strongly reduces the flexibility of the next one. This approach can result in a poor implementation for several reasons. Placement is performed on a set of pre-determined modules, thus the geometry and the internal structure of each module cannot be modified during this phase. A possibly advantageous configuration, for example one minimizing capacitive loading at critical nodes or maximizing interleaving within the modules, cannot be explored.

More recently, attempts to alleviate, in part, these limitations have been proposed. In [3] and [6] placement algorithms based on Simulated Annealing [7] were presented that perform *module abutment* as a standard annealing move. Moreover, in [3] the annealing operates on the shape of the modules to obtain the desired aspect ratio of the cell. However, the number of elemental modules is relatively large, and the configuration space becomes a limiting factor for the efficiency of this approach.

In this paper we propose the use of simultaneous placement and module optimization as an effective way to insure maximal flexibility during the placement phase, while drastically reducing the search space for all possible module implementations. First, the composite stack-module generator LDO partitions the circuit and finds different alternative sets of modules. Each alternative solution is chosen so as to minimize a cost function accounting for all analog-specific constraints. Routability and interconnect parasitics cannot be taken into account at this stage since no information on the reciprocal position of the modules is known. Next, all the equivalent solutions are made available to PUPPY-A , a Simulated Annealing based placement algorithm. The set of moves of the annealing algorithm has been extended to include not only geometric perturbations, but also swaps between alternative solutions. In this way, placement and module optimization are performed simultaneously. The set of modules available to the placement is relatively small, since the configurations yielding large performance degradation have already been discarded. Hence, negligible computational overhead is needed with respect to standard placement with a predefined set.

The paper is organized as follows. Section 2 describes the optimal module generation process. In Section 3 the placement algorithm is outlined and the techniques for integrating the optimization mechanisms are described. Section 4 shows the effectiveness of our approach through examples.

2 Optimum Module Generation

LDO [8] is a tool for MOS transistor composite stack generation. Its purpose is to generate a set of stacks containing all the transistors of the circuit, split into modules abutted with each other. Source/drain regions are shared between adjacent elements, in such a way that area and critical capacitances are minimized. Stack generation is performed by exploiting the equivalence between stacks and *paths* in the circuit graph. A path is a connected sub-graph whose vertices have either one or two adjacent edges. Every full-stacked implementation of the layout corresponds to a *path partition* of the circuit graph, namely a set $\mathcal{P}$ of paths, satisfying the *non-overlapping* condition, that no two stacks in the layout contain the same transistor, and the *covering* condition, that each transistor must appear in a stack. Notice that in every circuit at least one trivial partition exists, where each path has exactly one edge. Such partition corresponds to separate elemental transistor modules and it is often the starting configuration for placement tools with automatic abutment capability [3, 6]. The complexity of placement algorithms is considerably reduced if more complex stacked structures, with interleaved transistors and multiple abutments, are available as basic modules.

A *stack-generation algorithm* is used to determine the best partitions according to an optimality criterion based on a cost function. Constraints on parasitic junction capacitances, area, local routability, matching and symmetries can be taken into account simultaneously.

The Stack-Generation Algorithm

The stack-generation algorithm consists of two phases. In the first phase all the existing paths in the circuit are generated by a dynamic programming procedure. At the first step of this procedure, all paths of length 1 are generated. At the n-th step, for $n > 1$, all paths of length n are generated by augmentation of the paths of length $(n-1)$. Augmentation is carried out by adding to each path one of the edges connected to its endpoints. In the second phase the problem of path partition is transformed into a *clique* problem [9, p.194]. Every path becomes a vertex for a *path-graph* G_p, whose edges link two vertices if and only if the corresponding paths are *mutually compatible*, that is if they can coexist in the same partition. The latter condition holds if the non-overlapping rule, symmetry and matching constraints are all satisfied. The clique problem is solved with the iterative Bron-Kerbosch augmentation algorithm [10]. At the end of phase 2, the paths associated with the vertices of each clique satisfy the non-overlapping condition. If they satisfy the covering condition too, they form a partition.

The order in which different devices appear in a stack is relevant, since it affects parasitics, matching and symmetry constraints. Moreover, because of large transconductances, noise constraints and matching requirements, transistors with large W/L ratios are often needed. Hence, module splitting yields configurations where several devices are connected in parallel, and the number of edges is often larger than the number of vertices. The size of the clique problem becomes huge even with small circuits. In order to overcome these limitations, heuristics have been introduced enforcing analog constraints, such as matchings, symmetries and parasitic bounds. Costly or infeasible solutions of the stack generation problem are discarded early, thus effectively improving computational efficiency, while maintaining admissibility.

Analog Constraints and Computational Cost

The cost function exploits the fact that the junction capacitance of diffusion regions located in external positions of a stack is generally larger than that of internal regions. Capacitance can be minimized in critical nets by penalizing the nets located at the ends of a stack. The cost associated to a stack p is:

$$F(p) = \sum_i \text{cap}(n_i) \cdot \text{crit}(n_i),$$

where the sum is extended to all the nets n_i connected to the source/drain regions of stack p. Cap(n_i) is the junction capacitance of net n_i, while crit(n_i) is its criticality weight defined on the ground of the performance sensitivity with respect to this capacitance. As an example, consider the 2-module transistor shown in Figure 1.a and its two implementations 1.b and 1.c. If the capacitance on net S is more critical than that on net D, the cost of solution 1.c is lower than that of solution 1.b.

Symmetry constraints are effective in decreasing the computational cost of the algorithm. As soon as a path is found in the first phase of the algorithm, it is checked against symmetry constraints, and discarded if

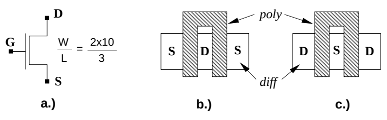

Figure 1: a.) A transistor split in two modules. b.) Layout minimizing the capacitance of net D. c.) Layout minimizing the capacitance of net S.

they are violated. The size of the clique problem is reduced accordingly, along with the overall CPU cost. Matching is accounted for by providing proper splitting of the transistors into modules with the same channel width and by abutting matched devices into the same stacks when possible. Matching can also be improved by selecting the configurations with maximum device interleaving, and common-centroid patterns are always found when they exist.

3 Placement Algorithm

A placement tool based on the Simulated Annealing algorithm, PUPPY-A [6], has been modified to implement our techniques of simultaneous optimization. In placement problems the annealing algorithm operates on an initial configuration by randomly perturbing it in order to minimize a cost function f by a finite succession of improvements. Every perturbation or *move* transforms a configuration s into a new one s' both S, the set of all possible layout configurations or *search space*. A move is accepted with probability 1 if it leads to a better configuration, i.e. a state with lower cost. Otherwise, the move is accepted with probability $P = e^{-\frac{\triangle F(s,s')}{T}}$, where $\triangle F(s,s')$ is the cost increase due to the move and T is a parameter called *temperature*. The value of T is determined at each point in time by a monotonically decreasing function or *cooling schedule*. With a sufficiently large number of iterations the algorithm converges to a configuration with a low cost function value [11].

In general, the cost function takes into account chip area and wiring constraints. In PUPPY-A analog specific requirements such as symmetry, matching, well distribution and parasitic constraints are also considered. The constraints are computed by PARCAR [4, 12] using constrained optimization based on a numerical sensitivity analysis of the circuit.

Modified Annealing Algorithm

In most placement algorithms S is fixed, since the aspect-ratio, the number and the internal structure of each module is pre-determined. The modules in the circuit are often generated **before** the placement phase and their shape and connectivity remains intact during the annealing. Slight deviations to this policy have been proposed in [3] and [6] where abutment and minor aspect-ratio adjustments were used to alter some modules during the annealing. In our placement approach the search space S is **dynamically modified** through a special move which swaps a randomly selected module with appropriate replacements. When a swap occurs the entire sub-circuit associated with the module is replaced with a new one, selected within a pool of all available alternatives which have been determined in advance during the optimum module generation phase. After the sub-circuit has been replaced the cost function f is reevaluated and the move is accepted or rejected according to the annealing scheme. At each swap only a small subset of highly optimized modules are considered as alternatives, thus insuring efficiency and robustness. In [13] this modified annealing scheme has been shown to converge under the same conditions of [11].

Alternative Circuit Realizations

To illustrate the advantages of considering alternative module implementations during the placement, let us analyze the simple circuit shown in Figure 2. For simplicity, assume all transistors are equal in

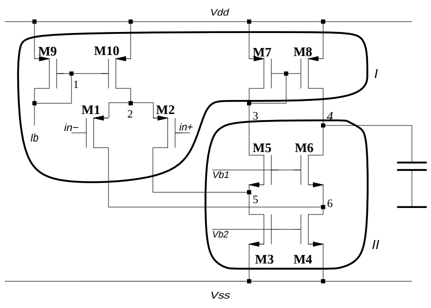

Figure 2: Example of a folded cascode opamp. The bubbles represent the sub-circuits created by the module generator on the ground of transistor polarity.

size. Moreover, assume that transistors ($M1$, $M2$), ($M3$, $M4$), ($M5$, $M6$), and ($M7$, $M8$) require (1) to be matched devices and (2) to be placed symmetrically with respect to a vertical axis. In addition, assume (3) that $M9$ and $M10$ also be a pair of matched devices.

This circuit can be partitioned in two sub-circuits according to the polarity of its transistors (See bubbles in Figure 2). If the transistors are implemented in a "full-stacked" design style, a possible solution for each sub-circuit could be the following scheme: ($M9$, $\frac{M10}{4}$, $\frac{M1}{2}$, $\frac{M1}{2}$, $\frac{M10}{4}$, $\frac{M7}{2}$, $\frac{M7}{2}$, $\frac{M8}{2}$, $\frac{M8}{2}$, $\frac{M10}{4}$, $\frac{M2}{2}$, $\frac{M2}{2}$, $\frac{M10}{4}$) for sub-circuit I, and ($M5$, $M3$, $M4$, $M6$) for sub-circuit II. The notation $\frac{Mxx}{n}$ indicates one of the n modules of width w/n into which transistor Mxx, of width w, is split. This scheme is desirable in terms of symmetry and matching constraints and it is acceptable in terms of area, since only two stacks implement the entire circuit. However, it has several drawbacks. Firstly, this solution does not allow any interleaving among transistors $M1$ and $M2$. This might result in worse offset and noise performance in presence of even modest technology gradients. Secondly, due to the size of the stack implementing sub-circuit I, nets 5 and 6 might be long and therefore involve stray resistances, large capacitances to ground and cross-coupling capacitances. This might result in poor bandwidth, due to the high criticality of these nets.

An alternative scheme for sub-circuit I, which could alleviate most of these problems, would be to distribute all devices in three stacks: ($\frac{M9}{2}$, $\frac{M10}{8}$, $\frac{M9}{2}$, $\frac{M10}{2}$) , ($\frac{M2}{2}$, $\frac{M1}{2}$, $\frac{M2}{2}$, $\frac{M1}{2}$) and ($\frac{M7}{2}$, $\frac{M1}{8}$, $\frac{M7}{2}$, $\frac{M8}{2}$) . This implementation of sub-circuit I is perfectly equivalent to the previous one, in terms of the cost function defined in Section 2. Although equivalent however, these alternatives may yield very different circuit performance, depending on the routing. In other words, no module generator would have enough information to select a solution among the two alternatives before a complete layout is actually placed and possibly routed. For this reason both alternatives must be made available to the placer in order to insure that the best possible realization be selected.

Module Replacement Criteria

The placement algorithm is responsible for finding the best possible combination of all available alternative realizations for a given circuit. At this point however the placer has better tools to perform this selection. In fact, it can make precise estimations of global wiring, parasitics (cross-over capacitance, stray resistance and capacitance, etc.)

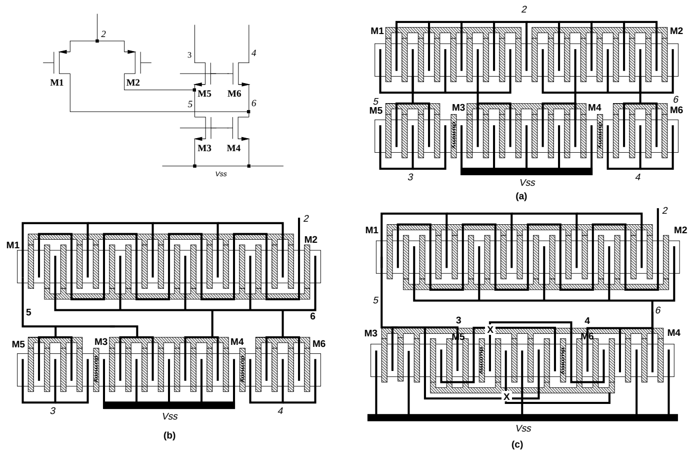

Figure 3: Alternative implementations of the differential pair and its active load.

and routability. Thus, since the cost function takes all these factors into account, the decision of accepting or rejecting the new alternative is supported by a much better insight.

Capacitive and resistive parasitics are estimated using analytical models [14] for a particular technology. Cross-over capacitances between nets are obtained from an estimate of the probability that the nets will cross after the routing as described in [6]. From these estimates the cumulative effect of parasitics is evaluated based on the sensitivity information of performance with respect to each parasitic component in the circuit as suggested by [4]. Violations to specifications can therefore be detected and included as an additional cost f_{co} in function f [6]. For a particular performance W_i, the component f_{co} is proportional to its deviation $\triangle W_i$ from nominal. $\triangle W_i$ is approximated using a linearization of performance W_i at its nominal point $\triangle W_i = \sum_j S_j^i \, p_j$. S_j^i is the sensitivity of performance W_i with respect to parasitic p_j and p_j represents any parasitic component or mismatch. At each temperature T the component f_{co}, along with global wiring, area, etc. , determines whether to accept the new configuration.

To illustrate the selection mechanism let us consider the input differential pair and its active loads from the circuit of Figure 2. Some implementations in a "full-stacked" design style are shown in Figure 3 a), b), c) . The first realization (a) shows minimum interleaving of all devices, this configuration represents the worst possible device matching for the pair of transistors $M1$-$M2$, $M3$-$M4$, and $M5$-$M6$. However interconnect parasitics are small and routing symmetry of nets 5, 6 is high.
In the third stack (c) considerably higher matching is achieved by configuring the transistor geometries according to a common-centroid pattern. This configuration requires however more wiring area and unavoidable signal path crossings (X) are introduced.
Configuration (b) is a trade-off between the previous two, with moderate interconnect length and good interleaving. Notice that no crossings are present in this configuration.

For simplicity, consider only nets 5, 6, Vss, devices $M1$, $M2$, $M3$, $M4$ and performances W_1, W_2. For given transistor sizes and bias current, using sensitivity analysis, deviations $\triangle W_i$ for W_i, $i = 1, 2$, can be approximated as

$$\begin{aligned}\triangle W_i = \quad & \alpha_{i1} \triangle R_{S\_21} + \alpha_{i2} \triangle R_{S\_34} + \alpha_{i3} R_{S\_1} + \alpha_{i4} R_{S\_2} \\ & \alpha_{i5} \triangle V_{t\_21} + \alpha_{i6} \triangle V_{t\_34} + \alpha_{i7} C_{56} + \alpha_{i8} C_5 + \alpha_{i9} C_6,\end{aligned}$$

where R_{S_j} and $\triangle R_{S\_jk}$ are the degeneration resistances and resistance mismatches at the sources of devices Mj and Mk. $\triangle V_{t\_jk}$ are the mismatches of voltage threshold in the device pairs Mj and Mk.

Parasitic capacitance C_{jk} represents the coupling between net j and k, while C_j is the substrate capacitances of net j.
Suppose the specifications for W_i are given in terms of the inequality

$$\triangle\ W_i \leq \overline{\triangle\, W_i},\ i = 1, 2, \tag{1}$$

where $\overline{\triangle\, W_i}$ represents the maximum acceptable deviation of performance W_i from nominal.

Let us consider now the three alternative implementations of the differential pair of Figure 3. These alternatives are all equivalent in terms of area and junction capacitances. However, realization (a) requires the smallest routing area for the interconnect of nets 5 and 6. This implies low interconnect resistances and capacitances. Moreover better matching between nets 5 and 6 are obtainable in the routing phase. Suppose now $\alpha_{i1}, \ldots, \alpha_{i4}, \alpha_{i7}, \ldots, \alpha_{i9}$ are large $\forall\ i$, i.e. resistive and capacitive mismatches dominate threshold voltage mismatches in affecting both performances. Then, there will be no specification violation in the sense of equation 1 and the contribution of f_{co} will be negligible or null. Otherwise, f_{co} will increase the cost of this configuration, thus decreasing the probability of its selection.

Consider now realization (c). This solution minimizes the threshold voltage mismatch, though at the expenses of the capacitive coupling and self capacitance of nets 5 and 6. So, if α_{i5} and α_{i6} are large $\forall\ i$, then the cost of this configuration will be lower.

If both specifications were tight, i.e. a strong dependence of both performances of parasitic mismatches and threshold voltages was present, a trade-off configuration should be chosen. For instance, (b) represents a possible alternative configuration that could meet both specifications. Under the above conditions, the cost of this configuration is in fact lower than that of the other two. Thus the probability of acceptance is the highest among all configurations.

Clearly, if no flexibility were allowed during the placement phase, it would not have been possible to enforce tight specifications on both performances. Hence, a constraint-driven approach to placement, in combination with module optimization is desirable, when many specifications, possibly tight, are present and trade-offs are possible.

4 Results

Several industrial circuits have been tested on the algorithm. Some of the most relevant examples are discussed in this section. All circuits were routed with ROAD and compacted with SPARCS-A [5]. Performance specifications were obtained by simulating the extracted circuit with HSPICE and SPICE33 .

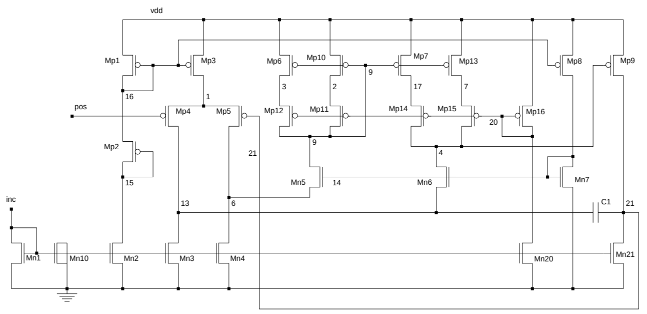

Figure 4: Schematic of "fcphil" (Courtesy of Philips Research Labs, Eindhoven, The Netherlands).

module generation (LDO)			
circuit	# transistors	# sub-circuits	# alternatives (per sub-circuit)
fcphil	40	9	36/8/1/1/1/1/1/1/1
mph	51	17	182/97/24/3/3/2/2/1/1/1/1/1/1/1/1/1/1

Table 1: Data on the module generation phase.

placement (PUPPY-A)			
	# states visited	# swaps	# acceptance ratio
fcphil	532,322	17,578	0.153
mph	365,565	21,494	0.068

Table 2: Data on the placement phase.

Performance	nominal	specification	measured
fcphil			
offset	2.03 mV	+/- 0.1 mV	2.05 mV
unity gain bandwidth	28.18 MHz	- 20kHz	28.91 MHz
low frequency gain	35.30 dB	- 0.2 dB	35.15 dB
phase margin	70.05 deg	+/- 5 deg	66.3 deg
mph			
power	435 μW	+/- 130 μW	543 μW
supply voltage	1.5 mV	+/- 0.15 mV	full range

Table 3: Specifications and performance of extracted layout.

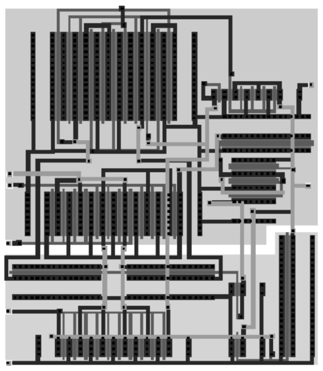

Figure 5: Final layout of "fcphil" .

Fast Folded Cascode Amplifier

Consider the folded cascode operational amplifier shown in Figure 4. The specifications are listed in Table 3. Sub-circuit partitioning and module generation are outlined in Table 1. Data regarding the placement are in Table 2. Module generation and placement required 1.5 seconds and 881.1 seconds respectively on a DECstation5000/125. Final layout and performance evaluation are shown in Figure 5 and in Table 3 respectively. All specifications, although tight, were met.

Micro-Power Amplifier

Consider now the amplifier depicted in Figure 6 (compensation has been omitted). Tight specifications on power and supply range (See Table 3) were imposed to the design. Table 1 lists the results of the module optimization phase. Data regarding the placement are listed in Table 2. Module generation and placement required respectively 3.3 and 6096.72 seconds on a DECstation5900/260. A Monte Carlo analysis was conducted on the circuit based on available statistical data of technological mismatches from a commercial 1 μm process. The resulting performance evaluation is shown in Table 3. Figure 7 shows the final layout.

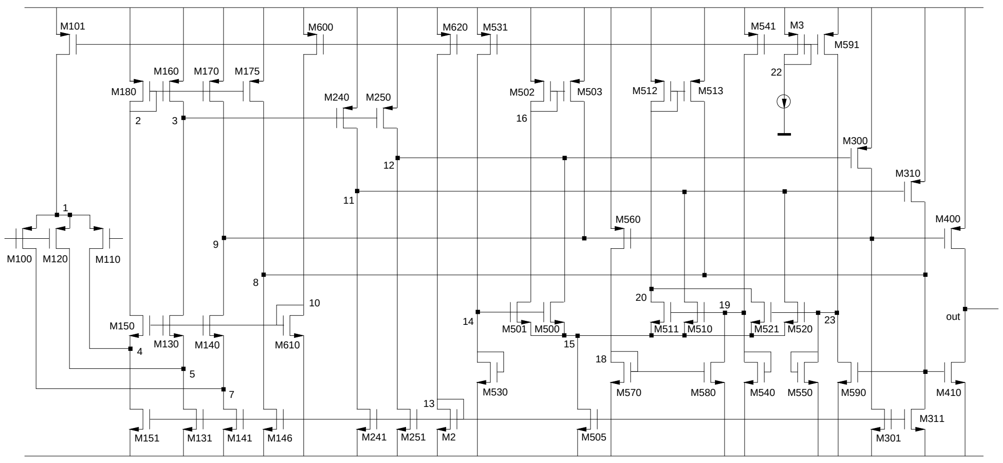

Figure 6: Schematic of "mph" (Courtesy of Delft University of Technology, Delft, The Netherlands).

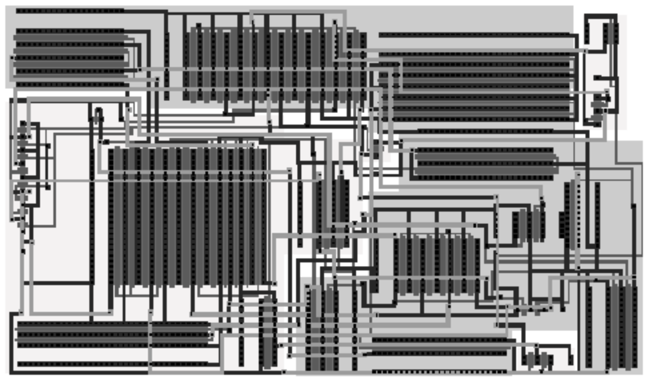

Figure 7: Final layout of "mph" .

5 Conclusions

A novel technique for simultaneous placement and module optimization of analog integrated circuits has been proposed. A module generator places all active devices in stacks. The modules are optimized for area in the respect of analog-specific constraints dictated by performance. When a number of equivalent optimal solutions is found, all solutions are made available to the placer. The placer, based on Simulated Annealing, operates the final implementation selection, with a better knowledge and a more accurate estimation of interconnect parasitics. Hence, more options can be considered during the placement phase and a more sophisticated selection mechanism can be used, thus insuring a robust approach to the enforcement of tighter performance specifications.

Acknowledgements

This research has been partially supported by DARPA and SRC. The first author has been supported by a grant of Asea Brown Boveri, inc. Baden, Switzerland. The authors would like to thank M. Pelgrom of Philips Research, J. H. Huijsing and R. G. H. Eschauzier of Delft University of Technology, The Netherlands for providing the examples presented in this work.

References

[1] J. Rijmenants, J. B. Litsios, T. R. Schwarz and M. G. R. Degrauwe, "ILAC: An Automated Layout Tool for Analog CMOS Circuits", *IEEE Journal of Solid State Circuits*, vol. 24, n. 2, pp. 417–425, April 1989.

[2] M. Kayal, S. Piguet, M. Declercq and B. Hochet, "SALIM: A Layout Generator Tool for Analog ICs", in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 751–754, May 1988.

[3] J. M. Cohn, D. J. Garrod, R. A. Rutenbar and L. R. Carley, "KOAN/ANAGRAM II: New Tools for Device-Level Analog Placement and Routing", *IEEE Journal of Solid State Circuits*, vol. 26, n. 3, pp. 330–342, March 1991.

[4] U. Choudhury and A. Sangiovanni-Vincentelli, "Constraint Generation for Routing Analog Circuits", in *Proc. Design Automation Conference*, pp. 561–566, June 1990.

[5] E. Malavasi, H. Chang, A. Sangiovanni-Vincentelli, E. Charbon, U. Choudhury, E. Felt, G. Jusuf, E. Liu and R. Neff, "A Top-down, Constraint-Driven Design Methodology for Analog Integrated Circuits", in *Analog Circuit Design*, pp. 285–324. J. H. Huijsing, R. J. van der Plassche and W. Sansen Ed., Kluwer Academic Press, 1993.

[6] E. Charbon, E. Malavasi, U. Choudhury, A. Casotto and A. Sangiovanni-Vincentelli, "A Constraint-Driven Placement Methodology for Analog Integrated Circuits", in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 2821–2824, May 1992.

[7] S. Kirkpatrick, C. Gelatt and M. Vecchi, "Optimization by simulated annealing", *Science*, vol. 220, n. 4598, pp. 671–680, May 1983.

[8] E. Malavasi, D. Pandini and V. Liberali, "Optimum Stacked Layout for Analog CMOS ICs", in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 1711–1714, May 1993.

[9] M. R. Garey and D. S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., New York, NY, 1979.

[10] C. Bron and J. Kerbosch, "Algorithm 457 - Finding all cliques of an undirected graph", *Comm. ACM*, vol. 16, n. 9, pp. 575–577, September 1973.

[11] F. Romeo, "Simulated Annealing: Theory and Applications to Layout Problems. PhD Thesis", Memorandum UCB/ERL M89/29, University of California at Berkeley, March 1989.

[12] E. Charbon, E. Malavasi and A. Sangiovanni-Vincentelli, "Generalized Constraint Generation for Analog Circuit Design", in *Proc. IEEE ICCAD*, pp. 408–414, November 1993.

[13] E. Charbon, E. Malavasi, D. Pandini and A. Sangiovanni-Vincentelli, "Imposing Tight Specifications on Analog IC's through Simultaneous Placement and Module Optimization", in *Proc. IEEE Custom Integrated Circuits Conference*, May 1994.

[14] U. Choudhury and A. Sangiovanni-Vincentelli, "An Analytical-Model Generator for Interconnect Capacitances", in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 861–864, May 1991.