Imposing Tight Specifications on Analog IC's through Simultaneous Placement and Module Optimization

Edoardo Charbon, Enrico Malavasi†, Davide Pandini† and Alberto Sangiovanni-Vincentelli

Department of Electrical Engineering and Computer Sciences, University of California, Berkeley
†Dipartimento di Elettronica e Informatica, Università di Padova, Italy

Abstract

Techniques are presented for simultaneous placement and module optimization for analog ICs. An algorithmic approach to module generation provides alternative sets of modules, optimized with respect to performance but with different trade-offs among area, parasitics and matching. A Simulated Annealing algorithm performs the placement, selecting among the available configurations the one that best fulfills all performance and geometric requirements. Compared to standard approaches, the flexibility of placement is considerably increased, thus allowing the enforcement of tighter specifications.

1 Introduction

In recent years layout design automation for analog integrated circuits has generated considerable interest due to the increasing complexity of the problems and the sophistication of proposed solutions. In order to cope with higher performance sensitivity to layout topology and interconnect parasitics in analog circuits, several techniques have been proposed [1, 2, 3].

A fundamental limitation of all these approaches is that physical design is performed with no explicit reference to the performance specifications of the circuit. Recently, performance-driven approaches to the layout of analog circuits have been presented [4]. In [5] a complete set of tools for the layout synthesis of analog circuits based on this principle was proposed. The main limitation of this approach was the partitioning of the analog layout synthesis problem into its traditional phases, namely module generation, placement, routing and compaction.

Although performance constraints can be enforced by a performance-driven approach to analog layout, the achievement of very tight specifications is a difficult task. The main reason is that the procedure is inherently sequential, thus limiting the flexibility of each phase. In addition, placement and routing generally operate on template-based modules, thus offering little structural flexibility. This approach is often unsatisfactory for two reasons. Firstly, by operating with pre-constructed modules, common-centroid patterns and interleaved configurations cannot be built. These configurations are useful to achieve compact layouts which are independent of technological gradients. Secondly, junction capacitances are not taken into account. By properly abutting source and drain regions in MOS transistors, critical capacitances can be effectively reduced, with beneficial effects on performance. Attempts to alleviate, in part, these limitations were proposed in [6], where placement and routing of critical nets are performed simultaneously. For efficiency reasons however only a small number of nets can be considered, thus severely limiting the suitability of the approach.

In this paper we advocate the use of simultaneous placement and module optimization as an effective way to improve flexibility during the placement phase while drastically reducing the search space for all possible module implementations. In our approach two optimizers are integrated into one placement tool as illustrated in figure 1. Using technology information, Parcar[4], a constraint generator, maps performance specifications onto a set of bounds on critical parasitics. These constraints are first used by the composite module generator Ldo [7], to partition the circuit and to find different alternatives for the set of modules implementing it. Each alternative solution is chosen so as to minimize a cost function accounting for all analog constraints. Then Puppy-A [8], a SA based placement algorithm, is used, whose set of moves has been extended to include swaps between alternative solutions.

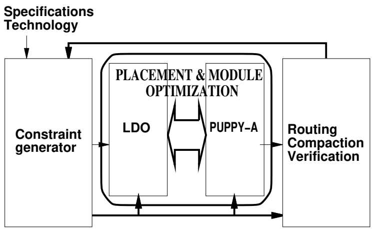

Figure 1*: Flow diagram of the tool.*

The cost function minimized by the annealing accounts for parasitic control and topological constraint minimization. Puppy-A performs placement and module optimization simultaneously, but only on those modules, which satisfy analog-specific constraints. Hence all performance specifications, if feasible, are guaranteed to be satisfied. The placed layout is then routed and compacted using tools based on a performance-driven methodology. The final layout is extracted and its performance is compared with the original specifications. In case specifications are not met, the parasitic constraints are regenerated and the entire process is carried out again.

The advantages of such an approach are manifold. A complete set of alternative optimal implementations is available to the annealing algorithm, thus insuring maximal flexibility. In addition, an early selection of module implementations purges the search space of all sub-optimal solutions which are guaranteed to worsen performance, thus increasing the efficiency of the search. Finally, routability and interconnect parasitics cannot be taken into account by Ldo since no information on the relative module positions is known at this stage. Therefore all the equivalent solutions in terms of area and parasitics are made available during the placement. Hence a better layout quality and higher robustness can be achieved.

The paper is organized as follows. In Section 2 the optimal module generation process is described. In Section 3 the placement algorithm is outlined and illustrated with simple examples. In Section 4 the practical suitability of our approach is demonstrated through circuits of industrial strength.

2 Optimum Module Generation

Ldo [7] is a tool for MOS transistor composite stack generation. All circuit transistors are split into modules abutted with each other to form stacks, thus sharing source/drain regions between adjacent elements. 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 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 $\mathcal{P}_0$ 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, 8].

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 are taken into account simultaneously. The stack-generation algorithm consists of two phases.

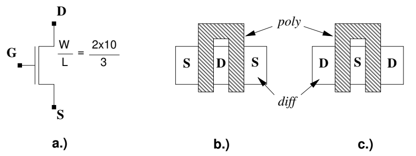

Figure 2: *a.) A transistor split in two modules. b.) Layout minimizing the capacitance of net D. c.) Layout minimizing the capacitance of net S.*

i) 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.

ii) In the second phase the problem of finding a path partition is transformed into a *clique* problem. 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.

In phase *ii,* The clique problem is solved with the Bron-Kerbosch iterative algorithm [9]. At each step of this algorithm, an attempt is made to augment a complete component. If augmentation is not possible, the complete component is maximal, i.e. it is a clique.

In [10] a *pairing criterion* was proposed to use this algorithm for a layout style, where n-type and p-type transistors are respectively aligned along parallel rows. However, in analog circuits n-type and p-type transistors are not usually balanced in number and size as they are in standard CMOS logic gates. Moreover, the order in which different devices appear in a stack is relevant, since it affects parasitics, matching and symmetry constraints. Finally, 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. Therefore the number of edges in the circuit graph is often big and the computational cost of the clique solver becomes unacceptable even with small circuits.

In Ldo these limitations have been overcome by introducing heuristics 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. The cost associated to a stack p is computed as follows:

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

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 2.a and its two implementations 2.b and 2.c. If the capacitance on net S is more critical than that on net D, the cost of solution 2.c is lower than that of solution 2.b.

At the end of phase 1 of the stack generation algorithm, all the paths found are checked against symmetry constraints. The paths violating such constraints are discarded and the size of the clique problem is reduced accordingly. 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.

Branch-and-bound has been applied to the complete-component augmenting step of the Bron-Kerbosch algorithm. A *predictor* provides an estimate $\tilde{F}$ of the cost F of the cheapest clique that can be generated by augmenting the complete component. If $\tilde{F}$ is larger than the cost of the cheapest clique found so far, augmentation is not carried out and the component is discarded. Admissibility is preserved if $\tilde{F}$ is an optimistic estimate, i.e. it is $\tilde{F} \leq F$ [7].

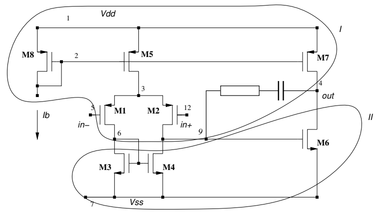

Figure 3: *Example of a two-stage opamp. The bubbles represent the partition operated by the module generator on the ground of transistor polarity.*

3 Placement Algorithm

Puppy-A [8] is a placement tool for analog custom modules based on SA. The original annealing algorithm has been modified to operate in parallel with the module optimizer. In particular, one of the four basic components which completely specify the algorithm [11, 12], the *move set*, has been extended.

The new move set includes simple moves such as translations, rotations and combinations of the two as well as moves involving two modules, such as swaps and abutments. Furthermore, moves for swapping alternative module implementations within the circuit has been added. The same moves are applied on special components, such as terminals and bias contacts. This is done to minimize performance degradation due to defective biasing which is a frequent source of latch-up and undesired body effects.

As an example, let us consider the simple circuit shown in Figure 3. For simplicity, but without loss of generality, assume all transistors are equal in size. Transistors $M1$, $M2$ and $M3$, $M4$ require (1) to be matched devices and (2) to be placed symmetrically with respect to a vertical axis. In addition, assume (3) that $M8$ and $M5$ 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 3). If the transistors are implemented in a "full-stacked" design style, a possible solution for each sub-circuit could be the following scheme: ($M1$, $\frac{M5}{2}$, $\frac{M8}{2}$, $\frac{M8}{2}$, $\frac{M5}{2}$, $M2$) and ($M7$) for sub-circuit I, and ($M3$, $\frac{M6}{2}$, $\frac{M6}{2}$, $M4$) for sub-circuit Note that $\frac{Mxx}{n}$ represents a transistor of width W split into n modules of width W/n each. This scheme is acceptable in terms of all matching and symmetry constraints and it is advantageous in terms of area, since only three stacks implement the entire circuit. However, this solution does not allow interleaving among transistors, which might result in a poor dynamic behavior in presence of even modest technology gradients. Moreover, due to the size of the stack implementing $M1$ and $M2$, nets 6 and 9 might be long and therefore involve stray resistances, large capacitances to ground and cross-coupling capacitances. This again might result in poor dynamic performance, due to the high criticality of these nets.

An alternative scheme is the following: ($\frac{M1}{2}$, $\frac{M2}{2}$, $\frac{M1}{2}$, $\frac{M2}{2}$) and ($M8$, $\frac{M5}{2}$, $\frac{M5}{2}$, $M7$). This implementation is equivalent to the previous one, in terms of the cost function defined in Section 2. However, it may yield different circuit performance, depending on the interconnect pattern of the circuit. 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. For this reason both alternatives must be made available to the placer in order to insure that the best possible realization be chosen.

The mechanism of selection for equivalent alternatives is mainly driven by routability and parasitic considerations. When the move "alternative solution swap" becomes active, a module in the circuit is picked at random and the implementation associated with it is swapped with an alternative one among those available for each sub-circuit. Estimates are made on the total wire length and on the number of unavoidable crossings. The cumulative effect of parasitics on performance is evaluated and the cost associated with it is computed as part of the general cost function. The new configuration implementing the chosen alternative solution is accepted based on the usual annealing scheme.

To illustrate the selection mechanism for alternative solutions let us consider the current mirror $M8-M5$ from Figure 3. Figure 4 shows implementations in a "full-stacked" design style. The first realization (a) shows maximum interleaving between the transistors of the pair. This configuration represents the best possible device matching for the pair considering technology gradients. The second stack (b) requires the smallest wiring area, at the expenses of matching. The last configuration (c) is a trade-off between the

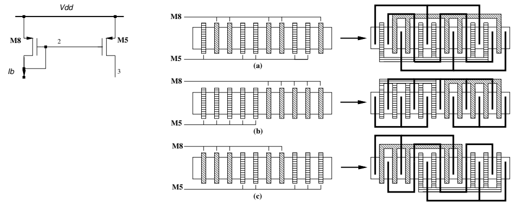

Figure 4: *Alternative implementations of the mirror $M8 - M5$.*

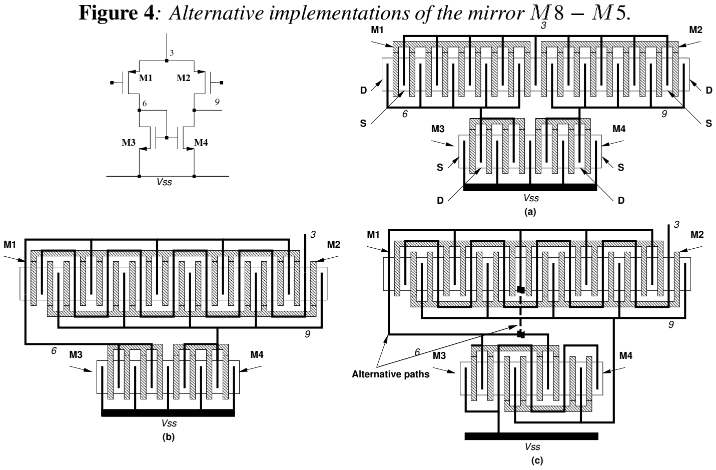

Figure 5: *Alternative implementations of differential pair and active load.*

previous two.

Suppose now that the matching requirements of the currents in the mirror were tight, as in the case of tight specifications on slew rate or gain, then the best candidates for selection would be (a) and (c). Hence, the final choice would be dictated essentially by global considerations, i.e. routability and parasitics of adjacent interconnect lines. If on the other hand, tight specifications on input capacitance at node 2 were imposed, then the configuration likely to be chosen would be (b) for its smaller routing area.

Consider now more complex structures, where the number of combinations becomes large. For the sake of clarity only a portion of the circuit in Figure 3 is quantitatively analyzed in terms of the effects on performance by interconnect parasitics. In Figure 5 the input differential pair and its active load are depicted. Consider two performances, namely *offset voltage (V_{off})* and *unity gain bandwidth (ω_1)*. With the use of sensitivity analysis, the degradation of both performances from their respective nominal value is computed as a linear combination of interconnect parasitics and parasitic mismatches. For simplicity but without loss of generality, consider the resistance and the capacitance toward substrate of nets 3, 6, 9 and all capacitive couplings between them. For given transistor sizes and bias current, using sensitivity analysis, the offset voltage degradation, $\triangle V_{off}$, can be approximated as

$$\begin{aligned}\triangle V_{off} = \; & -2.02\times10^{-5}\ \triangle R_{S\_21} - 1.33\times10^{-5}\ \triangle R_{S\_34} \\ & -5.00\times10^{-4}\ \triangle V_{t\_21} - 1.25\times10^{-4}\ \triangle V_{t\_34},\end{aligned} \quad (1)$$

where $\triangle R_{S\_21}$ and $\triangle R_{S\_34}$ are the mismatches between degeneration resistances at the source of transistors $M2, M1$ and $M3, M4$ respectively. $\triangle V_{t\_21}$ and $\triangle V_{t\_34}$ are the mismatches in voltage threshold in the differential pair and in the active load, respectively. The degradation of the unity gain bandwidth, $\triangle\omega_1$ is approximated as

$$\begin{aligned}\triangle\omega_1 = \; & 443.89\ \triangle R_{S\_34} + 461.99\, R_{S\_1} + 510.361\, R_{S\_2} - 1207.09\, C_{69} \\ & -1028.49\, C_6 - 842.685\, C_9,\end{aligned} \quad (2)$$

where $\triangle R_{S\_1}$ and $\triangle R_{S\_2}$ are the degeneration resistance at the source of transistors $M1$ and $M2$. Parasitic capacitance C_{69} represents the coupling between net 6 and 9, while C_6 and C_9 are the substrate capacitances of net 6 and 9, respectively. Suppose that a specification of 2.4V ($\pm$ 10 %) for the offset and of 6.6 MHz ($+\infty$/-100kHz) for the bandwidth is given. Using the constraint generator Parcar, a set of constraints for each parasitic element is generated. Let us consider now three alternative implementations of the differential pair of Figure 5, depicted in Figure 5. These alternatives are all equivalent in terms of area and junction capacitances.

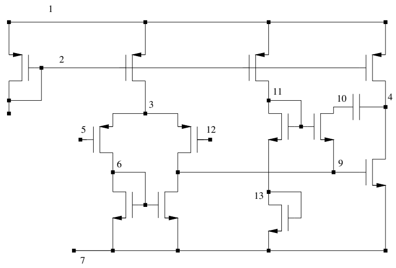

Figure 6: *Schematic of "opamp1".*

Realization (a) requires the smallest routing area for the interconnect of nets 6 and 9. This implies low interconnect resistances and capacitances and therefore low deviations of unity gain bandwidth. Estimates of parasitics would yield a $\triangle\omega_1$ of -74 kHz and a $\triangle V_{off}$ of 0.3 mV . Clearly only one specification can be met with such a configuration, unless the specification on the offset is relaxed. This is mainly due to the role played by the threshold voltage mismatches.

Consider now realization (c). This solution minimizes the threshold voltage mismatch, though at the expenses of the capacitive coupling and self capacitance of nets 6 and 9. In fact the the performance ω_1 exceeds specifications (-173.98 KHz), while V_{off} becomes acceptable (150.33 μV).

If both these tight performance specifications must be enforced simultaneously, a trade-off configuration should be chosen. For instance, (c) represents the only configuration allowing both specifications to be satisfied. In fact $\triangle V_{off}$ and $\triangle\omega_1$ are 148.0 μV and 97.2 KHz , respectively.

If no flexibility were allowed during the placement phase, it would not have been possible to enforce such tight constraints, thus limiting the suitability of the approach.

Convergence Properties

SA has been proved to converge for the class of problems to which placement belongs provided a set of conditions. In [12, 13] conditions were set on the number of moves per temperature, $t_k \rightarrow \infty$. A stronger result, proved, among others, by [14], guarantees convergence for $t_k = 1$, and $T_k = a/ln\ k$, with a sufficiently large.

Claim: **The same conditions for convergence of [12] hold for the proposed placement scheme.**

The annealing algorithm with the proposed modification is still a Markov Chain. Since all new states added to the search space are all reachable, and the same properties apply to these states as to the others, the chain is regular and recurrent. Since, for fixed temperature, the transition rule to go from a state to the next is invariant, the chain is also homogeneous. Hence, the standard theory of homogeneous Markov Chains applies. As a result, at a given temperature all states of the chain converge to a stationary probability distribution (Theorems 3.1.4 and 3.2.1 in [12]). This implies that at each temperature a global minimum is reached, thus the convergence of the algorithm follows (Propositions 3.2.1, 3.2.2 and Theorem 3.2.2 in [12]). □

Similar arguments can be used to prove convergence in the sense of [14], thus ensuring that the whole approach is robust.

4 Results

Several industrial-strength circuits have been tested on the proposed algorithm. Some of the most relevant examples to this discussion are described in detail hereafter.

Consider the two-stage amplifier of Figure 6. The initial specifications and nominal values are listed in Table 3. Firstly, a set of alternative implementations for each circuit partition was created using Ldo . Table 1 shows the results of this phase. The CPU time was 0.5 seconds on a DECstation 5000/125. Secondly, Puppy-A was applied to the circuit with a CPU time of 224.88 seconds on a DECstation 5000/125. Table 2 lists results of the placement phase. The layout in Figure 7 is the final result after the routing and the compaction phase. The circuit was extracted using the analog-specific extractor Estpar and simulated with SPICE3 . The measured performance is described in table 3. All specifications were met.

module generation (Ldo)		
# basic transistors	# sub-circuits	# alternatives
56	8	16/6/2/2/1/1/1/1

Table 1: *Data on the module generation phase.*

placement (Puppy-A)		
# config. visited	# implementation swaps	acceptance
21,600	1,708	25.3%

Table 2: *Data on the placement phase.*

Performance	nominal	specification	measured
unity gain bandwidth	5.15 MHz	$+\infty$/-50 kHz	5.63 MHz
phase margin	65 deg	+/- 5 deg	61 deg
low frequency gain	70.0 dB	+/- 1 dB	70.0dB
offset	0.225 mV	+/- 0.225 mV	0.405 mV

Table 3: *Specifications and performance of extracted layout.*

module generation (Ldo)		
# basic transistors	# sub-circuits	# alternatives
40	9	36/8/1/1/1/1/1/1/1

Table 4: *Data on the module generation phase.*

placement (Puppy-A)		
# config. visited	# implementation swaps	acceptance
532,322	17,578	15.3 %

Table 5: *Data on the placement phase.*

Performance	nominal	specification	measured
offset	2.03 mV	+/- 0.3 mV	2.025 mV
unity gain bandwidth	28.183 MHz	- 20kHz	28.250 MHz
low frequency gain	35.304 dB	- 0.1 dB	35.529 dB
phase margin	70.0 deg	+/- 5 deg	65.0 deg

Table 6: *Specifications and performance of extracted layout.*

Consider the folded cascode operational amplifier shown in Figure 8. The specifications are in Table 6. The sub-circuit partitioning and module generation is outlined in Table 4 and the placement data are in Table 5. The module generation and the placement required 1.5 seconds and 881.1 seconds on a DECstation5000/125. The final layout and the performance evaluation are shown in Table 6 and in Figure 9, respectively. Note the tightness of the specifications, which were all met.

5 Conclusions

A novel technique for simultaneous placement and module generation targeted towards circuits with extremely tight specifications has been presented. A module generator partitions the circuit based on analog specific considerations. Then, for each sub-circuit it finds a set of optimal modules implementing it, by solving a clique problem on a graph representing all possible combinations of feasible modules. The placer, based on Simulated Annealing, operates the final implementation, with a better knowledge and a more accurate estimation of interconnect parasitics, thus insuring robustness and flexibility.

Acknowledgements

This research has been partially supported by DARPA and SRC. The authors would like to thank M. Pelgrom of Philips Research, The Netherlands, for providing some of 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.

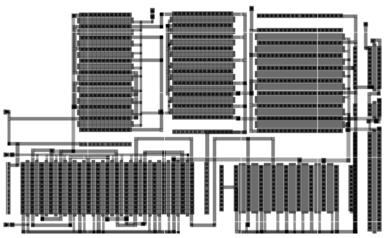

Figure 7: *Final layout of "opamp1".*

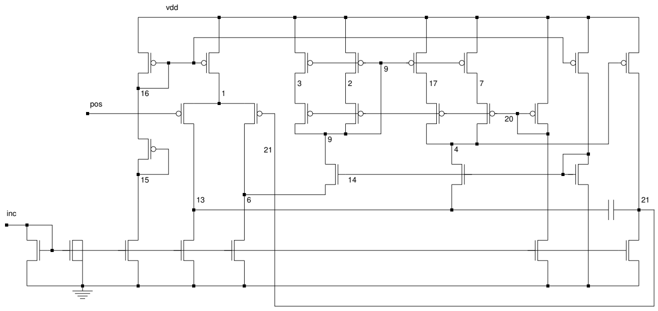

Figure 8: *Schematic of "fcphil" (Courtesy of Philips Research Labs, Eindhoven, The Netherlands).*

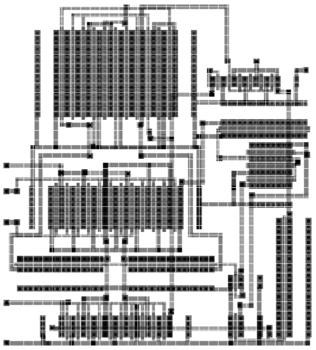

Figure 9: *Final layout of "fcphil" .*

[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. DAC*, 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] J. M. Cohn, D. J. Garrod, R. A. Rutenbar and L. R. Carley, "Techniques for Simultaneous Placement and Routing of Custom Analog Cells in KOAN/ANAGRAM II", in *Proc. ICCAD*, pp. 394–397, November 1991.

[7] 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.

[8] 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.

[9] 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.

[10] S. Wimer, R. Y. Pinter and J. A. Feldman, "Optimal Chaining of CMOS Transistors in a Functional Cell", *IEEE Trans. on CAD*, vol. CAD-6, N.5, pp. 795–801, September 1987.

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

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

[13] M. Lundy and A. Mees, "Convergence of the Annealing Algorithm", *Mathematical Programming*, vol. 34, pp. 111–124, 1986.

[14] B. Hajek, "Cooling Schedules for Optimal Annealing", *Mathematics of Operations Research*, vol. 13, n. 2, pp. 311–329, May 1988.