Template-Driven Parasitic-Aware Optimization of Analog Integrated Circuit Layouts*

Sambuddha Bhattacharya, Nuttorn Jangkrajarng and C-J. Richard Shi
Department of Electrical Engineering, University of Washington
Seattle, WA, 98195-2500
{sbb, njangkra, cjshi}@ee.washington.edu

ABSTRACT

Layout parasitics have great impact on analog circuit performance. This paper presents an algorithm for explicit parasitic control during layout retargeting of analog integrated circuits. In order to ensure desired circuit performance, bounds on layout parasitics' magnitudes are determined first. Then, graph techniques are coupled with mathematical programming to constrain layout geometry based on these parasitic bounds. The algorithm has been demonstrated to ensure desired circuit performance during technology migration and performance specification changes.

Categories and Subject Descriptors

J.6 [**Computer Applications**] Computer-Aided Engineering – *computer-aided design*.

General Terms: Algorithms, Performance, Design.

Keywords: Analog Layout Automation, Parasitics, Sensitivity, Optimization.

1. INTRODUCTION

Layout symmetry, device floorplans, relative placement and layout parasitics are of immense importance in ensuring desired analog circuit performance [1]. Layout parasitics arise from the transistor source/drain capacitances, interconnect resistances, and line and coupling capacitances. These parasitics can have significant impact on circuit performances such as gain, bandwidth, and phase margin. However, they cannot be accurately estimated before a layout is actually completed. This presents a major challenge to analog layout automation [2].

Recently, automatic template-based layout generation is emerging as an effective solution to analog layout automation. In the template-based analog layout automation tool IPRAIL (Intellectual Property Reuse-based Analog IC Layout) [3], an optimized layout is automatically generated from a symbolic structural template that contains device floorplan, symmetry, matching, and wiring alignment information. The templates can be extracted automatically from a coarse-grained layout generated by macro-cell based methods [2] or from an existing fine-tuned silicon-proven layout manually crafted by designers [3]. By automatically extracting and reusing the designers' knowledge embedded in an existing layout, IPRAIL has demonstrated its efficacy for technology migration and electrical performance specification changes.

* This research has been supported in part by the U.S. DARPA's NeoCAD program and in part by the NSF's ITR program.

DAC 2005, June 13–17, 2005, Anaheim, California, USA.

In this paper, we present how template-based techniques can be explored for parasitic-aware optimization of analog circuit layout. A key observation is that the structural templates define the layout geometry in terms of constrained layout variables, therefore permit the accurate modeling of layout parasitics *parametrically* in terms of these layout variables. Then constraints on layout parasitics can be enforced during layout generation.

With this, parasitic-aware analog layout generation is solved in three steps: (i) Identification of limits on parasitic values that guarantees desired performance, using the sensitivity of circuit performance to parasitics. (ii) Determination of constraints on layout geometric variables due to the parasitic bounds. (iii) Generation of target layout in presence of geometric constraints due to parasitics, symmetry, relative placement and design rules by combining graph-based methods and linear programming (LP).

2. PROBLEM DEFINITION

2.1 Layout Retargeting

Layout retargeting refers to the generation of a target layout from an existing layout [3]. In this method, a set of constraints corresponding to technology design rules, layout symmetry etc. is first extracted from the input layout. These constraints force the target layout to retain floorplan, symmetry and other properties of the existing layout. The objective of retargeting is to generate the target layout with minimum area while obeying these constraints.

Layout retargeting can be formulated as a one-dimensional compaction problem [3]. If x_R and x_L are the right and left ends of a layout, the problem in horizontal direction is given as:

$$\begin{aligned} &min && (x_R - x_L) && (1.1)\\ &subject\ to && x_i - x_j \geq const \quad , \quad x_i - x_j = const && (1.2)\\ & && x_i - x_j = x_k - x_l && (1.3) \end{aligned}$$

where all variables correspond to the left and right edges of the layout rectangles. The constraints in Eqs. (1.2) and (1.3) correspond to design rules, fixed device widths and symmetry.

2.2. Parasitic Aware Layout Retargeting

Parasitic aware layout retargeting refers to the generation of a target layout from an existing layout such that the layout parasitics in the target layout are maintained within acceptable limits. These restrictions on parasitic values impose geometric constraints. The interconnect parasitics can be estimated with simple expressions for resistance and capacitance given as $r = (l/w).r_{SH}$ and $c = 2.l.C_{SW} + l.w.C_A$ where r_{SH} is the sheet resistance, C_{SW} and C_A are the sidewall and area capacitances and l and w are the length and width of the wire. The coupling capacitances are given as a linear function of $1/d$ where d is the distance between two wires.

Parasitic aware layout retargeting refers to the problem defined in Eq. (1) with additional geometric constraints due to parasitics. Clearly, the parasitic models are nonlinear in terms of geometry and depend on both the horizontal and vertical dimensions of the wires. However, for superior computational speed, the geometric

constraints due to parasitics need to be linearized in both horizontal and vertical dimensions to the form in Eq. (2).

$$x_i - x_j \geq const \quad , \quad x_i - x_j \leq const \tag{2.1}$$
$$x_i - x_j = x_k - x_l \tag{2.2}$$

Eq. (2.1) refers to geometric lower and upper bounds due to parasitics. Eq. (2.2) arises for matched interconnects.

2.3 Methodology

We incorporated our algorithm into IPRAIL. The steps in constraint generation are shown in Fig. 1(a). First, all transistors, passive devices and nets are extracted from the input layout. The design rule, connectivity, coupling and symmetry constraints are extracted next. Determination of bounds on parasitic values and generation of geometric constraints are described in Section 3.

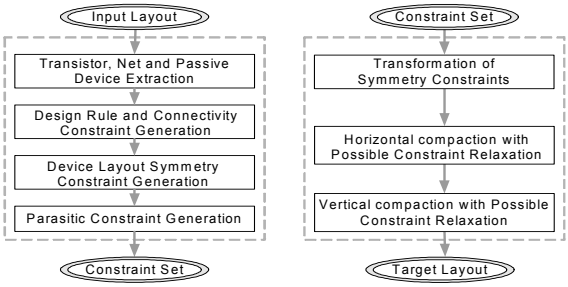

Fig. 1: (a) Steps in constraint generation for retargeting. (b) Steps in target layout generation from the imposed constraints.

Layout generation, shown in Fig. 1(b), is a generalized compaction problem. Due to superior running time of graph-based solution for the compaction problem compared to LP, the constraint equations are converted into a graph form. The constraints in Eq. (1.2) and (2.1) are directly convertible into the graph form where the variables represent the nodes of the graph and the constant on the right hand side denotes the weight of the arc connecting two nodes. The symmetry constraints in Eq. (1.3) and matching constraints in Eq. (2.2) cannot be directly imposed into the graph as no "right-hand constant" is known a-priori. These constraints are transformed into a graph imposable form by a combination of graph-based longest path algorithm and LP [4].

3. PARASITIC CONSTRAINT GENERATION

The steps for computing geometric constraints due to parasitics are shown in Fig. 2. First, parasitics are extracted from the input layout. The sensitivity of the circuit performance parameters to the parasitics is computed next. Circuit sensitivities, parasitics, and the performance parameters are related by a set of linear constraints. We formulate a geometric programming (GP) problem with these constraints to maximize the bounds on the values of parasitics. These bounds are then mapped to the layout geometric constraints.

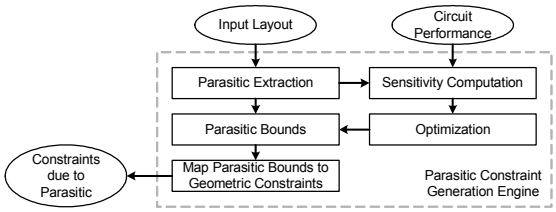

Fig. 2: Flow of parasitic dictated geometric constraints generation.

3.1. Extraction of Input Layout Parasitics

The interconnect parasitics in the input layout offer guidelines for constraining the parasitics in the target layout. The resistive and capacitive parasitics are expressed in terms of length, widths and spacings of the wires. These geometries are in turn stated in terms of the variables associated with the edges of the layout rectangles. Our parasitic extractor expresses resistances, capacitances and coupling capacitances in terms of these variables.

3.2. Sensitivity-Based Bounds for Parasitics

Numeric bounds on the parasitics are computed such that the parasitic values below these bounds would guarantee desired circuit performance [5]. Consider a set of circuit performances $\{ F_i \}$, where $i=1..M$, and a set of parasitics $\{ P_j \}$ where $j=1..N_P$. Let $\{ F_{i\text{-}nom} \}$ be the set of nominal values of the performance parameters defined in absence of layout parasitics. Our objective is to generate bounds on the set of parasitics $\{ P_j \}$ of the form.

$$P_K = P_L \tag{3}$$
$$P_K \leq P_{K\text{-}Bound} \tag{4}$$

Eq. (3) corresponds to matched parasitics and Eq. (4) represents maximum bound on a parasitic. The parasitic values are bounded such that a change ΔF_i in the performance from the nominal value lies within an acceptable range. This is mathematically defined as

$$\Delta F_i \leq \Delta F^{+}_{i-\max} \quad \forall \quad F_i \geq F_{i-\text{nom}} \tag{5}$$
$$\Delta F_i \geq -\Delta F^{-}_{i-\max} \quad \forall \quad F_i \leq F_{i-\text{nom}} \tag{6}$$

The generation of constraints on parasitic values is based on the sensitivity of the performance parameters with respect to the parasitics in the circuit. The sensitivity of a performance parameter F_i with respect to a parasitic element P_j at $F_{i\text{-}nom}$ is defined as

$$S_{ij} = \left[\partial F_i / \partial P_j \right]_{P_j = 0} \tag{7}$$

From the maximum allowable deviations of the performance parameters and the sensitivity of the performance parameters to the parasitics, a set of linear constraints are generated as follows.

$$\sum_{j=1}^{N_P} S^{+}_{ij} P_j \leq \Delta F^{+}_{i-\max} \quad \forall F_i \geq F_{i-\text{nom}} \tag{8}$$

$$\sum_{j=1}^{N_P} S^{-}_{ij} P_j \leq \Delta F^{-}_{i-\max} \quad \forall F_i \leq F_{i-\text{nom}} \tag{9}$$

where $S^{+}_{ij} = S_{ij}$ if $S_{ij} \geq 0$ and $S^{+}_{ij} = 0$ if $S_{ij} < 0$, $S^{-}_{ij} = -S_{ij}$ if $S_{ij} \leq 0$ and $S^{-}_{ij} = 0$ if $S_{ij} > 0$.

Larger parasitic bounds result in geometric constraints that are easier to be satisfied during layout generation. This computation of bounds on the parasitics is modeled as a GP problem of the form

$$min \quad P_1^{-\alpha_1} P_2^{-\alpha_2} ... P_N^{-\alpha_N} \tag{10}$$

subject to the linear constraints in Eq. (8) and (9). Here, the α_i are positive constant weights given according to the relative magnitude of the corresponding parasitics in the input layout.

3.3. Linearized Geometric Constraints Generation

Once the bounds on each individual parasitic values are obtained, they need to be translated to geometric constraints on each section of the wires. Consider the interconnect wire shown in Fig. 3(a). Let R_{1B} and C_{1B} be the maximum resistance and capacitance of its leftmost section. Then the following equations relate the width and lengths of the section to the parasitic bounds.

$$x_2 - x_1 \leq (R_{1B} / r_{SH}) . (y_4 - y_3) \tag{11}$$
$$x_2 - x_1 \leq C_{1B} / (2C_{SW} + C_A . (y_4 - y_3)) \tag{12}$$

The width and length of the section and the bounds on the sections resistance and capacitance define a parasitically feasible region shown shaded in Fig. 3(b). In addition, there is a range of length and width of the wire that would allow the target layout to be constructed. This geometrically feasible region is illustrated with the

dotted area. Our objective is to define the geometric upper bounds due to parasitics so as to maximize the overlap between geometrically feasible and parasitic feasible regions.

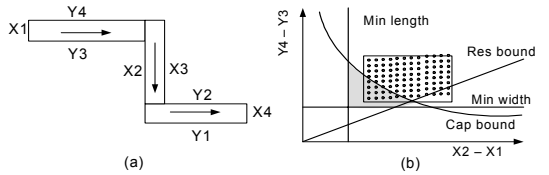

Fig. 3: (a) Sections of a wire with current directions. (b) The feasible region defined by the resistive and capacitive bounds for the leftmost horizontal section of the wire is shaded. The dotted region represents the corresponding geometrically feasible region.

Initial geometric feasible region for each wire section in the horizontal direction is obtained by two runs of longest path algorithm on the constraint graph, from the left to the right and from the right to the left. At this stage, the constraint graph does not include the parasitic constraints. The minimum widths are estimated based on the locations of the layout rectangles.

The generation of constraints on dimensions of wires requires fitting the largest rectangle in the overlap regions of geometric and parasitic feasibility. Let L_i and W_i be the lengths and widths of the i^{th} wire section. Let R_B and C_B be the bounds on the resistance and capacitance of the corresponding net and L_{i_min} and W_{i_min} be the geometrically feasible minimum sizes. Then the maximum overlap between geometric and parasitic feasible regions can be formulated as the following GP problem and solved with the optimization library of [6].

$$min \quad \sum (L_i . W_i)^{-1} \tag{13.1}$$

$$subject\ to: \quad \sum r_{SH} . L_i / W_i < R_B \tag{13.2}$$

$$\sum (C_A . L_i . W_i + 2\, c_{SW} . L_i) < C_B \tag{13.3}$$

$$L_i > L_{i\_min}, \quad W_i > W_{i\_min} \tag{13.4}$$

4. LAYOUT GENERATION

After the generation of the geometric upper and lower bound constraints due to parasitics, a horizontal and a vertical constraint graph are constructed with constraints due to design rules, connectivity, symmetry and parasitics. Here, we explain our algorithm with the horizontal constraint graph. If all the horizontal constraints are feasible, then the longest path algorithm can be employed directly to solve the compaction problem [3]. Upon its completion, the longest path algorithm finds the x-positions of the left and right edges of all layout rectangles. However, some constraints due to the parasitics may be infeasible because of two reasons. First, the geometric feasible region for a wire section may change due to parasitics in other layout rectangles. Second, the bounds on the parasitics may be too tight to be accomplished.

Geometric upper bounds due to parasitics of the form of Eq. (2.1) show up as negative weight arcs from the node for the right rectangle edge to the node for the left rectangle edge. These arcs, along with design rule left-to-right arcs can produce cycles. Positive cycles occur for small negative arc weights that render the sum of all arc weights of the cycle greater than zero. The longest path algorithm fails to terminate in presence of positive cycles as the layout dimension increases to infinity.

The resolution of positive cycles from the constraint graph is essential for the modified compaction problem. Resolution refers to reassignment of the constraint weights such that the sum of all arc weights is zero or less. We employ a combination of longest path and LP based constraint refinement to solve this problem. First, the algorithm identifies the positive cycles during a longest path run and then extracts the constraints due to parasitics in the positive cycle. It then resolves the positive cycle by refining the constraints due to those parasitics. The algorithm is shown in Fig. 4. Here, care needs to be taken so that resolution of one positive cycle does not introduce other positive cycles in the graph. This is based on the following observation.

Obs.: Increasing the magnitude of a negative arc weight does not introduce a positive cycle in other parts of the constraint graph.

We increase the magnitude of the negative arc weights corresponding to parasitics in order to resolve the positive cycles. This amounts to increasing the geometric upper bound due to parasitics. We proceed by resolving one positive cycle at a time by reassigning parasitic-dictated geometric constraints. This is accomplished by formulating an LP problem.

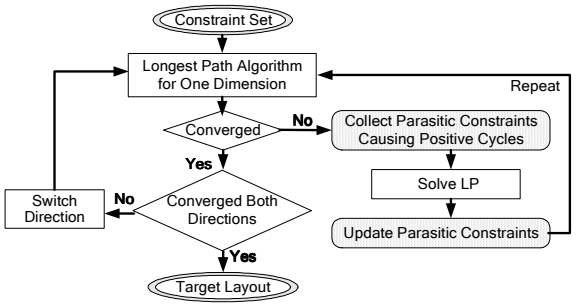

Fig. 4: Algorithm for layout generation with parasitic constraints.

Consider a positive cycle in the horizontal constraint graph with N negative weight arcs due to parasitics. Let T_i represent the variable corresponding to the i^{th} negative weight arc. Let R_i and C_i be the resistance and capacitance of the corresponding wire section. The reassignment of arc weights can be formulated as the following LP problem.

$$\text{min} \quad \sum \alpha_i R_i + \beta_i C_i \tag{14.1}$$

$$\text{subject to:} \quad \sum T_i > Positive\_cycle\_weight \tag{14.2}$$

$$T_i < K_i R_i \tag{14.3}$$

$$T_i < L_i C_i \tag{14.4}$$

where T_i, R_i and C_i are the decision variables while α_i, β_i, $K_i = y_{Width} / r_{SH}$ and $L_i = 1/(2c_{SW} + c_A . y_{Width})$ are constants.

The constraint graph is updated with the new weights obtained upon optimization and the algorithm is applied iteratively until all positive cycles are resolved. After the resolution of all positive cycles, the longest path algorithm settles to find the exact positions and sizes of all layout rectangles in the horizontal direction. The algorithm is then applied on the vertical constraint graph.

After the longest path algorithm settles in both directions, the parasitic values in some wire sections may still be above the bounds. This can be refined by relaxing these parasitic bounds by allocating some 'unused' parasitics from other wire sections.

5. RESULTS

The algorithm has been incorporated into IPRAIL. We present the results of parasitic-driven retargeting on a two-stage Miller-compensated operational amplifier (opamp) shown in Fig. 5 and a single-ended folded cascode opamp shown in Fig. 6. The opamps were designed initially in TSMC 0.25um CMOS technology and retargeted to TSMC 0.18um with new specifications.

The parasitics that affect circuit performances are indicated in Fig. 5 and Fig. 6. The bounds on the parasitic resistance and capacitance

were obtained from sensitivity-based optimization and are listed in the 2nd columns of Table 1. With these bounding values for the parasitics, the target layouts for the two-stage and folded-cascode opamps were obtained through parasitic-aware retargeting (PAR). The layouts were also retargeted without parasitic considerations (RWOP) for comparison. Parasitic values extracted from the target layouts for the respective cases are shown in Table 1. Here, the resistance values include metal, contact and gate-poly resistance. For multi-terminal nets, we report the sum of the parasitics of the component two-terminal nets. PAR achieves parasitic bounds for both designs.

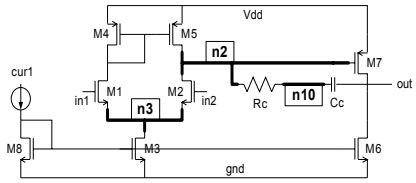

Fig. 5: Two-stage opamp: Nets n2, n3 and n10 are parasitic-sensitive.

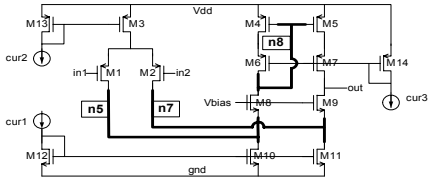

Fig. 6: Folded-cascode opamp: Nets n5, n7, n8 are parasitic- sensitive.

Table 1: Parasitic bounds obtained from sensitivity analysis, and parasitic values measured from layouts obtained by PAR and RWOP.

Two-stage Opamp				Cascode Opamp			
	Bounds	PAR	RWOP		Bounds	PAR	RWOP
R2 (Ω)	142.0	135.04	230.52	R5 (Ω)	53.3	36.25	36.39
R3 (Ω)	40.0	33.65	36.58	R7 (Ω)	57.6	38.38	38.57
R10(Ω)	51.4	49.08	286.9	R8 (Ω)	114.0	85.65	130.81
C2 (fF)	40.0	6.42	7.83	C5 (fF)	8.0	4.12	4.39
C3 (fF)	98.2	4.32	5.01	C7 (fF)	9.4	5.09	5.14
C10 (fF)	143.0	6.01	7.32	C8 (fF)	19.6	6.43	11.98

Table 2: Performances of layouts obtained by PAR and RWOP. The two-stage opamp layout obtained by RWOP has poor performance.

Layout		Gain (dB)	BW (MHz)	PM (°)	GM (dB)	Power (mW)	Area (um^2)
Two-stage Opamp	Perf. Thresh	62	100	90	10	-	-
	PAR	64.14	101.24	94.93	11.17	3.446	2684
	RWOP	64.02	361.34	69.32	6.08	3.445	2621
Cascode Opamp	Perf. Thresh	60	60	60	10	-	-
	PAR	62.61	63.48	60.06	10.29	0.877	2313
	RWOP	62.60	63.41	60.05	10.40	0.87	2313

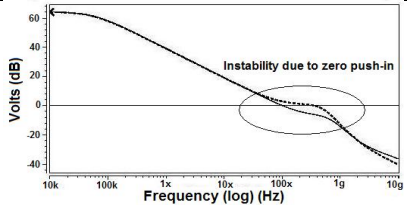

Fig. 7: Gain-bandwidth plot for two-stage opamp. The dotted curve is obtained for the layout generated by RWOP. The compensation zero is pushed inside the unity-gain frequency and leads to stability issues.

The simulation results for the two designs obtained with netlists extracted from the layouts are shown in Table 2. PAR achieves the desired specifications for both designs. Retargeting of the cascade opamp starts with a good template resulting in minor effects due to parasitics. The two-stage design shows significant differences in the stability measures for the layouts obtained by PAR and RWOP. For the two-stage layout obtained by RWOP, a zero is pushed within the unity-gain bandwidth leading to a comparatively unstable design as shown in Fig. 7.

Table 3: PAR and RWOP statistics for two-stage and cascode opamps.

	Two-stage Opamp		Cascode Opamp	
	PAR	RWOP	PAR	RWOP
# Nodes in Constraint-Graph	954		1020	
Design Rule, Sym Constraints	7398		9914	
Parasitic Constraints Arcs	176	0	116	0
Pos. Cycles Resolved - hor /ver	2 / 10	-	0 / 4	-
Template Extraction Runtime	4.9 s		5.9 s	
Layout Generation Runtime	10.1 s	4.6 s	9.0 s	4.5 s

The two-stage opamp layout obtained by PAR is shown in Fig. 8. Table 3 reports the statistics on the number of constraint graph nodes, number of design rule, symmetry and parasitic arcs for the two-stage and cascode opamps for PAR and RWOP. The two-stage opamp requires 12 positive cycle resolutions after generation of initial geometric constraints due to parasitics. The cascode opamp requires 4 positive cycle resolutions in PAR. For both designs, the target layout is generated within15 seconds of CPU time.

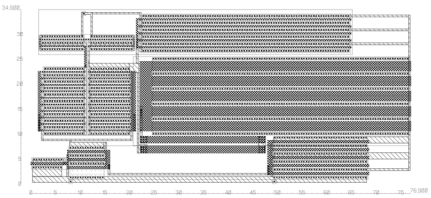

Fig. 8: Two-stage opamp layout generated by PAR.

6. CONCLUSIONS

In this paper, we presented an algorithm that enables explicit parasitic control in template-based retargeting of analog layouts. Bounds on layout parasitic values are obtained based on sensitivity analysis of circuit performance. Geometric programming is then employed to map the bounds on parasitic values to constraints on layout geometry. The target layout is then generated by an iterative longest path method with potential refinement of the parasitic dictated geometric constraint through linear programming. The algorithm has been incorporated into a computer-aided design tool called IPRAIL. This has been employed to retarget several analog layouts across technologies in a few seconds of CPU time.

REFERENCES

[1] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw Hill, 2001.

[2] K. Lampaert, G. Gielen and W. Sansen, "A performance- driven placement tool for analog integrated circuits", *IEEE Jour. Solid State Circuits*, vol. 30, pp. 773-780, Jul. 1995.

[3] S. Bhattacharya, N. Jangkrajarng, R. Hartono and C.-J. R. Shi, "Correct-by-construction layout-centric retargeting of large analog designs", *Proc. IEEE/ACM Design Automation Conference,* Jun. 2004, pp. 139-144.

[4] R. Okuda, T. Sato, H. Onodera and K. Tamaru, "An efficient algorithm for layout compaction problem with symmetry constraints", *Proc. IEEE/ACM Int. Conference on Computer-Aided-Design*, Nov. 1989, pp. 148-151.

[5] U. Choudhury and A. Sangiovanni-Vincentelli, "Use of performance sensitivities in routing analog circuits", *Proc. IEEE Int. Symp. Circuits and Systems*, vol.4, May 1990, pp. 348-351.

[6] Mosek manual, http://www.mosek.com/documentation.html.