Automatic Analog Layout Retargeting for New Processes and Device Sizes

Nuttorn Jangkrajarng, Sambuddha Bhattacharya, Roy Hartono, and C.-J. Richard Shi

Department of Electrical Engineering, University of Washington
Seattle, WA 98195 USA
{njangkra,sbb,rhartono,cjshi}@ee.washington.edu

ABSTRACT – *This paper presents an automatic analog layout resizing tool that can generate a new layout incorporating the target technology process and the target transistor sizes. The tool automatically preserves the analog layout integrity by extracting layout symmetry and matching, and then solving the constrained layout generation problem using a combined linear programming and graph-theoretic approach. The tool has been applied successfully to integrate specified transistor sizes and to migrate layouts for various analog designs from TSMC 0.25um CMOS to TSMC 0.18um CMOS process with comparable performances to re-design.*

1. Introduction

The scaling of feature size in VLSI circuits, both digital and analog, has been one of the strongest driving forces toward the rapid development of electronics technology. For digital circuits, the shrinkage of transistor sizes (for example from 0.25um to 0.18um to 0.13um) is the main reason microprocessors rapidly increase in speed. In the analog or mixed signal layouts, the circuit performances also benefit from smaller minimum feature size.

When there is a change in technology process, digital designers can utilize benefits from the new technology without much effort by using existing high-level VHDL or Verilog designs, scalable cell libraries, and readily available automatic place & route tools to generate a new circuit layout with better performances, or by layout retargeting tools. In contrary, analog designers do not have the comparable ability, which means they have to go through a full time-consuming cycle of redesigning, testing and drawing layouts. Therefore, an automated tool for re-layout of analog circuits will be essential in significantly reducing the design time for mixed-signal circuit technology migration.

In this paper, we present, for the first time, a computer-aided design tool that can automatically resize an existing analog layout for some modestly new processes. The tool is developed based on the existing algorithms [2-5] related to layout compaction. Its interface is shown in figure 1. The automatic analog layout resizer reads an original layout and its technology file, a new target technology file, and new transistor sizes. Then, it automatically generates a new layout that satisfies all the design rules while preserving all the analog layout integrities such as matching and symmetry constraints. The methodology we propose here is based on "recycling" already fine-tuned analog layouts. As high performance analog circuits are layout-sensitive and require device/wiring alignment, matching and symmetry, the method of recycling the layouts will be able to conserve the above requirements. Moreover, it will preserve all the unique aspects intended by engineers on any particular layout.

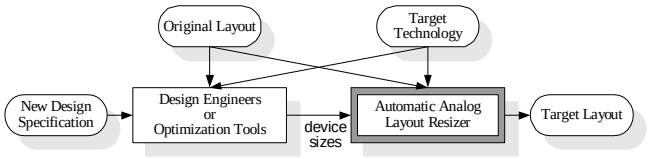

Figure 1: Interface of the automatic analog layout retargeting tool.

This paper is structured as follows. Section 2 presents the proposed analog layout retargeting flow and the implementation of important sections. Section 3 describes results using this new retargeting tool. Section 4 concludes the paper.

2. Automatic Layout Resizing Procedure

In order to automatically retarget an analog layout to a new technology process, three main considerations - namely new technology restriction, new device sizes, and layout structure preservation - have to be taken into account. The original design is used as a starting point for our program, with the purpose of maintaining the layout property. Our approach consists of layout representation and extraction [1,2], symmetry detection [3], technology conversion, circuit components resizing, and layout compaction [2,5]. The flow is shown in Figure 2, which important sections are described as follow.

2.1 Layout Representation and Symmetry Detection

First, the layout is represented by the corner stitching data structure [1], which recognizes each rectangle and its neighbors for every layer. The transistors and nets are then extracted from the layout.

The symmetry axes can be detected automatically between transistor pairs, based on the algorithm from [3]. However, this method may create many unnecessary symmetry axes. To overcome this problem, we introduce a user-specified option, which instructs the detection function to check and compare only

* This research was supported by NSF-ITR and DARPA NeoCAD grant.

between specified transistors or blocks. The symmetry axes then become one of the main criteria that the retargeting tool has to maintain.

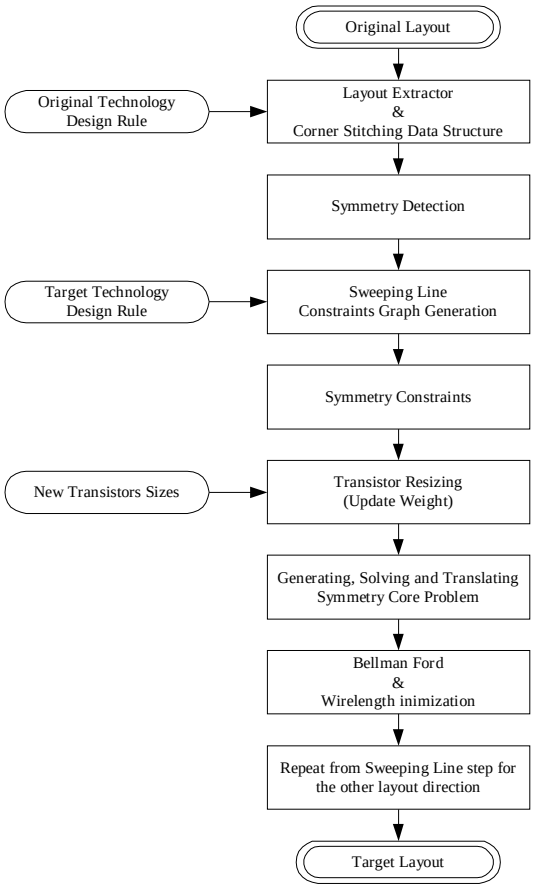

Figure 2: Proposed analog layout resizing flow.

2.2 Technology Migration Constraints

In order to facilitate the layout technology process migration and device resizing, a constraint graph that consists of nodes (representing the rectangles edges) and directed-weighted arcs (representing the constraints between edges) has to be created. One way of obtaining the graph is by using the sweeping line method [2]. First, the design rules for the target technology have to be acquired. Here, we categorize the design rules into three groups – minimum width, spacing, and extension. The sweeping line will start from the most left edge of the layout. While the line is traversing to the right, all required constraints from the current edge to the visible edges on its left will be added. The sweeping line algorithm also reduces redundant constraints, thus speeding up the solution solving. The example of constraints generated is shown in Figure 3.

As the sweeping line algorithm preserves the layout structure, our resizing tool requires two conditions: the target technology that covers all layers employed in the original layout, and the design that can be retargeted to the new process. Therefore, we shall call such process a *modestly* new process.

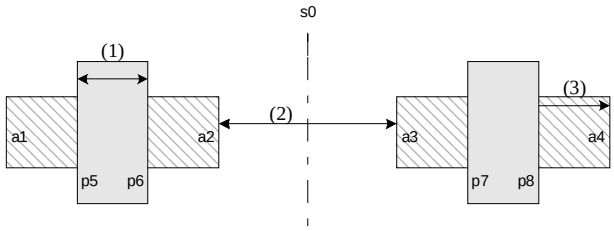

Figure 3: A layout of two transistors of only active layer (stripes) and poly layer (gray). Shown in numbers are examples of design rules: (1) minimum-width (2) spacing and (3) extension. The letters denote edges of active (a) rectangles and poly (p) rectangles, where (s) is a symmetry axis. Here are the constraints generated from this figure in horizontal direction. (Example design rules are taken from TSMC 0.25um)

a2-a1≥3	***a4-a3≥3***	***p6-p5≥2***	***p8-p7≥3***
p5-a1≥3	***a2-p6≥3***	***p7-a3≥3***	***a4-p8≥3***
a3-a2≥3	***s0-p6=p7-s0***	***p6-p5=p8-p7***	

2.3 Transistor Resizing

Typically, it is necessary to resize transistors to accomplish the same or better performance when a design is targeted on a new process. The target transistor sizes can be found either by manual simulations or, preferably, by some automatic transistor sizing tools.

Transistor resizing is accomplished by adding to the constraint graph the fixed-width constraints for each transistor to reflect new widths (added to active rectangles) and lengths (added to poly rectangles). This needs to be done on both horizontal and vertical direction. For symmetric transistor pairs, all the pairs have to be resized with exactly the same dimensions.

While performing transistor resizing, there is one difficulty regarding the number of active to metal-one contacts. Since the transistors sizes can be either tightened or widened, when decreasing transistors width, the reduced active area may not be able to fit all existing contacts. Thus a contact removal scheme has to be executed. After the addition of size-constraints, all active to metal-one contacts that reside by the transistors are removed. We, then, need to add two constraints between the metal-one rectangle edges and active rectangle edges, as shown in Figure 4, in order to preserve the connectivity between the two areas. After the constraint graph problem is solved, rows of contacts will be placed back in the right location.

2.4 Constrained Layout Generation

The new layout can be achieved by solving the constrained linear programming problem, which can be mathematically described as

$min\ (x_{r,o} - x_{l,o})$	*subject to layout constraint*
(a) $x_i - x_j \geq w$	*min-width, spacing, extension*
(b) $x_i - x_j = w$	*fixed-width*
(c) $x_i - x_j = 0$	*connectivity*
(d) $x_i - sym = sym - x_j$	*symmetry*

where variables $x_{r,0}$, $x_{l,0}$, and x_i or x_j are the most-right edge of the layout, the most-left edge of the layout, and any rectangles edges respectively.

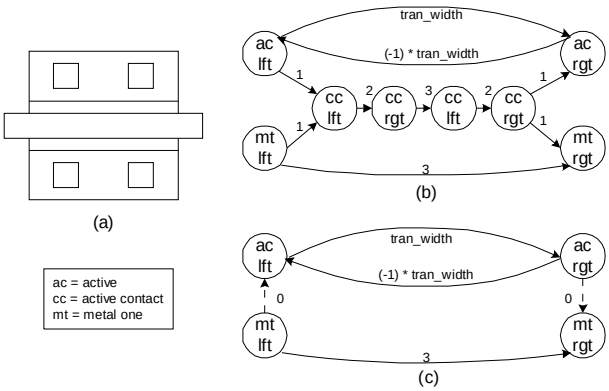

Figure 4: Contact removal in transistor resizing. (a) Transistor layout. (b) Original constraint graphs for only lower drain/source side with active contacts. (c) After removing contacts, two constraints are added for connectivity (metal->active and active->metal). Note: Constraints weights are taken from TSMC 0.25um process and in unit of lambda.

If we ignore the symmetry constraints, the above linear programming problem can be converted to the shortest path problem of the constraint graph represented by nodes as the layout rectangle variables and directed-weighted arcs as the design rule constraints.

From Section 2.2, constraint (a) $x_1 - x_2 \geq w$ can be expressed with a directed arc of weight w from node x_1 to x_2. Constraint (b) and (c) can be divided into two constraints of $x_1 - x_2 \geq w$ and $x_2 - x_1 \geq -w$, which also can be specified in the graph. Without the symmetry constraints, the minimum distance from one side of the layout to the other can be solved quickly with a graph-based shortest path algorithm (i.e. Bellman-Ford [6]).

However, in the presence of symmetry constraints, the linear programming is still necessary. Okuda *et al*. [4] has established an algorithm to solve this problem more efficiently by using advantages from both fast graph-based and linear programming technique. Instead of employing linear programming on a large problem, a smaller equivalent problem consisting of only the layout boundary variables and variables associated with symmetry axes are generated, and then solved with linear programming. The solution will give us the exact location of all variables in the equivalent problem. After that, we can replace each symmetry constraints with two fixed-width constraints and solve the compaction problem with the graph-based shortest path algorithm. Examples are

$$x_1 - sym = sym - x_2 \quad \Rightarrow \quad x_1 - sym = b \;\; and \;\; sym - x_2 = b$$
$$x_3 - x_4 = x_5 - x_6 \quad \Rightarrow \quad x_3 - x_4 = c \;\; and \;\; x_5 - x_6 = c$$

One weakness of the basic shortest path algorithm is that it tries to find the minimum distance from every variable to the starting (most-left) variable, which creates excessive leftward extension in some rectangles. It consequently results in bad performances due to surplus parasitic resistance and capacitance. Therefore, after solving the problem, minimization of individual rectangles as described in [2] or [5] has to be performed to secure good performance.

3. Examples

This section presents the results of applying our resizing tool on a single-ended folded-cascode and a two-stage operational amplifier. Both circuits are initially designed using the TSMC 0.25um CMOS process, and laid-out manually using Cadence's Virtuoso with multi-finger transistor structures. The target technology is the TSMC 0.18um CMOS process.

The CIF format files and the Cadence format technology file are imported from Virtuoso to our program. Once resizing is finished, the target CIF layout is exported back to Virtuoso, and a design-rule checking is performed. Finally, the netlists from both layouts are simulated by Hspice to compare their functionalities and performances.

3.1 Folded Cascode Operational Amplifier

Figure 5 shows the schematic of a folded cascode operational amplifier with 14 transistors (43 transistors if each finger is counted as a separate transistor). The original layout in the TSMC 0.25um process is shown in Figure 6, where the three symmetry axes A, B and C represented are {M1}:{M2}, {M3}:{M13} and {M4,M6,M8,M10}:{M5,M7,M9,M11}.

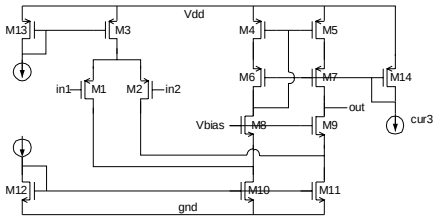

Figure 5: Schematic of a single-output folded-cascode opamp.

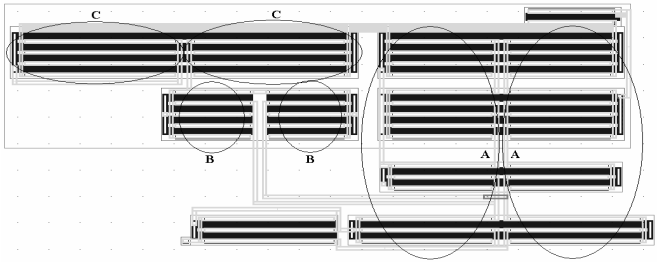

Figure 6: Original layout of the folded cascode operational amplifier in TSMC 0.25um. A, B and C are transistors symmetry blocks.

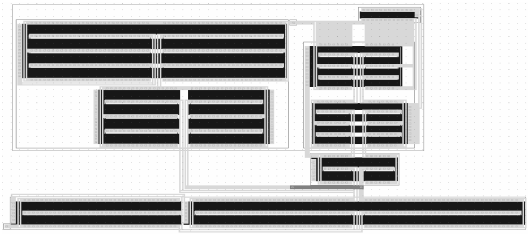

Figure 7: Resized layout of folded cascode opamp in TSMC 0.18um.

Figure 7 shows the resized layout in the TSMC 0.18um process. The transistor sizes are selected such that general operational amplifier specifications are met. The statistics on the performances and silicon area of the resized layout are summarized in Table 1, where "resize" and "no-resize" represent, respectively, the results with resized (the modified width and length from design engineers) transistors and no-resized (the same width and length as in the original layout) transistors.

Table 1: Performances comparison of folded-cascode opamp.

	0.25um	*0.18um no-resize*	*0.18um resize*
Vdd	*2.5 V*	*1.8 V*	*1.8 V*
Load Cap.	*1.0 pF*	*0.7 pF*	*0.7 pF*
Gain	*60.9 dB*	*61.9 dB*	*60.6 dB*
Bandwidth	*51.7 MHz*	*71.7 MHz*	*63.5 MHz*
Phase Margin	*63 deg*	*42 deg*	*71 deg*
Gain Margin	*12.5 dB*	*12.4 dB*	*10.5 dB*
Power	*1.48 mW*	*1.07 mW*	*0.88mW*
Area	*4,813 um^2*	*3,000 um^2*	*2,083 um^2*

3.2 Two-Stage Operational Amplifier

The schematic of a two-stage operational amplifier, shown in Figure 8, comprises of 1 MOS capacitor and 8 transistors (48 as each finger counted). There is one symmetry axis between two pairs of transistors {M1,M4}:{M2:M5}. The original layout (on TSMC 0.25um) is illustrated in Figure 9. The resizing is performed on width and length of all transistors, including the MOS capacitor. The target layout (in TSMC 0.18um) is depicted in Figure 10. The statistics on the performances and silicon area of the resized layout are summarized in Table 2.

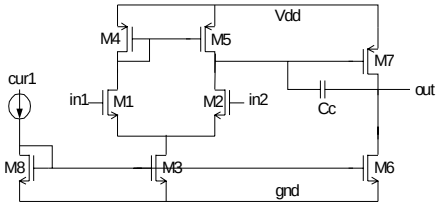

Figure 8: Schematic of a two-stage opamp.

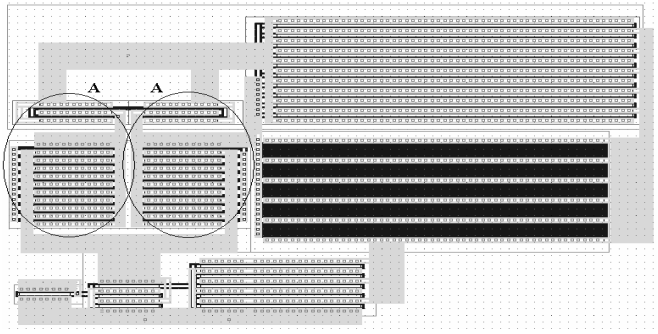

Figure 9: Original layout of the two-stage opamp in TSMC 0.25um.

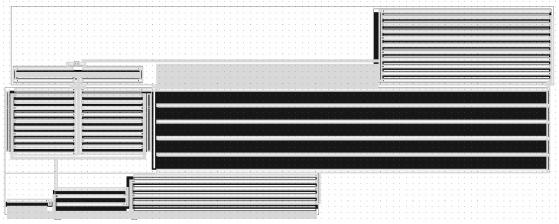

Figure 10: Resized layout of the two-stage opamp in TSMC 0.18um.

Table 2: Performances comparison of two-stage opamp.

	0.25um	*0.18um no-resize*	*0.18um resize*
Vdd	*2.5 V*	*1.8 V*	*1.8 V*
Load Cap.	*1.0pF*	*0.7pF*	*0.7pF*
Gain	*57.7 dB*	*39.6 dB*	*64.4 dB*
Bandwidth	*135 MHz*	*181 MHz*	*104 MHz*
Phase Margin	*50 deg*	*56 deg*	*56 deg*
Gain Margin	*9.6 dB*	*12.5 dB*	*9.2 dB*
Power	*4.82 mW*	*3.56 mW*	*3.46 mW*
Area	*3,648 um^2*	*2,664 um^2*	*2,745 um^2*

The runtime for the folded cascade opamp is 39.2 seconds and the two stage opamp is 37.6 seconds on a 440MHz SUN ultrasparc10 workstation.

4. Conclusion

An automatic tool for re-targeting existing analog layouts to new technology processes and new device sizes is presented. Layout recycling with symmetry detection and layout conservation scheme is applied in order to preserve the properties of analog circuit layout. Additionally, the tool has the ability to consider new transistor sizes to achieve better performances as part of the re-targeting process.

5. References

[1] J. K. Ousterhout, "Corner stitching: A Data-Structuring Technique for VLSI Layout Tools", *IEEE Transactions on Computer Aided-Design of Integrated Circuits and Systems*, pp.87-100, January 1984.

[2] S. L. Lin and J. Allen, "Minplex – A Compactor that Minimizes the Bounding Rectangle and Individual Rectangles in a Layout", *Proceedings of Design Automation Conference*, pp.123-130, 1986.

[3] Y. Bourai and C. J. R. Shi, "Symmetry Detection for Automatic Analog Layout Recycling", *Proceedings of Asian and South Pacific Design Automation Conference*, pp.5-8, 1999.

[4] R. Okuda, T. Sato, H. Onedera and K. Tamaru, "An Efficient Algorithm for Layout Compaction Problem with Symmetry Constraints", *Proceedings of International Conf. on Computer Aided Design*, pp.148-151, Nov. 1989.

[5] G. Lakhani and R. Varadarajan, "A Wire-Length Minimization Algorithm for Circuit Layout Compaction", *Proceedings of International Symposium on Circuits and Systems*, pp.276-279, 1987.

[6] T. H. Cormen, C. E. Leiserson and R. L, Rivest, *Introduction to Algorithms*, MIT Press, 1990.

Acknowledgement: The authors would like to thank Kiyong Choi and Jinho Park, SOC lab, Dept. of Electrical Engineering, University of Washington, for valuable discussions on circuit examples.