Multiple Specifications Radio-Frequency Integrated Circuit Design with Automatic Template-Driven Layout Retargeting

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 layout retargeting tool that generates analog and RF layouts incorporating new device sizes and geometries based on new circuit specifications. A graph-based symbolic template is automatically constructed from a practical layout such that expert designer knowledge embedded in the layout is preserved. The template can be solved for multiple layouts based on different device sizes and geometries, satisfying several different specifications. Symmetry conservation and passive device modification are also embedded in the tool. The retargeting tool is demonstrated on a voltage controlled oscillator to generate three layouts with different target goals. While manual re-design is known to take days to finish, the automatic layout retargeting tool takes a few hours to generate a reusable template and takes minutes to generate comparable layouts.

I. Introduction

The ability to integrate digital, analog and radio frequency (RF) circuits on to the same silicon chip, known as *system-on-chips* or *SOC*, has revolutionized the semiconductor industry. The portability and economy that results from integrating multiple functions on a single chip, nevertheless, is accompanied with an escalating complexity. This, together with the added pressure of aggressive design cycle, not only demands innovation in the field of computer-aided-design (CAD), but also necessitates the adoption of the design-reuse philosophy.

Continued advances in the CAD tools and the cell-based design methodology have already addressed these issues in the design of digital circuits. Unfortunately, CAD tools for analog/RF design still await major innovations. Indeed, design reuse in the analog/RF domain is often limited to only the circuit topology. Significant trade-offs between the major design goals like gain, bandwidth, stability, noise reduction, linearity and power minimization necessitate considerable amount of re-design. In addition, analog/RF circuit performance is strongly affected by layout styles and layout designers often need to use their expertise to eke out the required design specifications.

Fortunately, significant progress has been made recently in the form of optimization tools that synthesize analog/RF circuits for target specifications [1]. Automatic layout generation based on optimizations coupled with floorplanning, placement and routing of pre-designed macro-cells has also been reported in [2] and [3]. Despite their effectiveness and generality in obtaining desired circuits in various specification ranges, these layout automation schemes require extensive computation and at times fail to incorporate the expertise of the layout designer. Therefore, these methods are seldom adopted in the industry.

On the other hand, *template based* methods that require designer involvement provide a viable alternative. A high-quality template, created once, can be reused for multiple layout generation under different specifications. One such approach based on template generation with the Virtuoso Parameterized Cell tool has been proposed in [4]. Founded on the same principle of design reuse, another approach of automatically retargeting analog layouts was presented in [5]. While, the template creation in [4] requires substantial effort from the user and is very time-consuming, the method in [5] presents a scheme for automatic generation of a structural template from an existing layout. In this method, an already fine-tuned layout is used to automatically create a *symbolic structural template* incorporating the floorplan, symmetry and device/wiring alignment information. The new device sizes under changes in performance specification are imposed on the template, and the layout is realized by layout compaction with symmetry constraints [6].

In this paper, we propose, for the first time, a template based layout retargeting tool for RF integrated circuits. While being based on the same general principle as [5], this work adds substantial innovations in numerous aspects. Firstly, unlike analog circuits, RF design requires extensive handling of passive devices. Changes in specification of RF circuits require major modifications in the shapes, structures and sizes of on-chip spiral inductors, capacitors, and resistors. The RF layout retargeting method proposed in this paper handles such changes in shapes and sizes of passive devices. Secondly, RF layouts operating at gigahertz frequencies oftentimes incorporate innumerable number of vias for performance requirements. A naive template creation with hundreds of thousands of vias is extremely computationally intensive. This work provides a novel scheme for reduction of template size in the presence of such large number of vias. Thirdly, an automatic symmetry detection scheme is also employed to preserve device matchings. Finally, the automatic symbolic template created in the process can be used to generate multiple high-quality RF layouts for different design specifications.

The rest of the paper is organized as follows. Section II discusses the overall retargeting methodology and the symbolic structural template. Innovation in automatic symmetry detection, via/contact removal, and passive device retargeting are explained in Sections III, IV, and V respectively. Section VI presents the result of the retargeting tool on the voltage controlled oscillator. Section VII concludes the paper.

* This research has been supported in part by the U.S. Defense Advanced Research Projects Agency's NeoCAD program and in part by the National Science Foundation's ITR program.

II. Analog and RF Layout Retargeting via Structural Symbolic Template

The proposed method for automatically retargeting analog and RF layout is based on a *recycling* scheme. Fig. 1 illustrates the flow and interface of a structural template based layout retargeting tool. First, a structural symbolic template is constructed from an already fine-tuned layout. The template contains circuit topology, connectivity, design rules, placement and matching information of the layout. Next, new sets of *device sizes*, obtained through simulation, for each target specification are imposed on the structural template. Finally, target layouts for each specification are obtained by solving the enhanced templates.

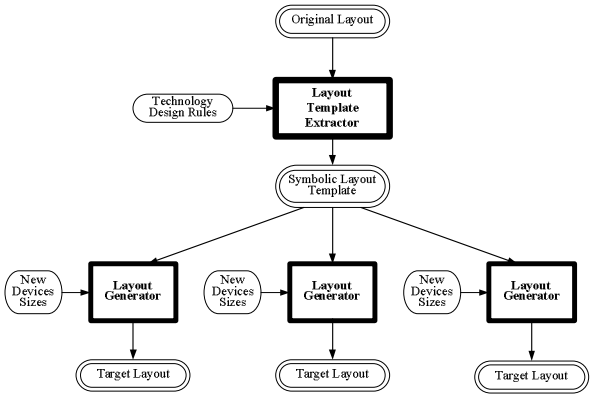

Fig. 1: Flow of the symbolic template-based layout retargeting tool.

A. Layout template Extraction

The main tasks of the layout template extractor are to identify active and passive devices, to detect device matching, and to assemble a structural symbolic template. Fig. 2 shows specific tasks of the layout extractor.

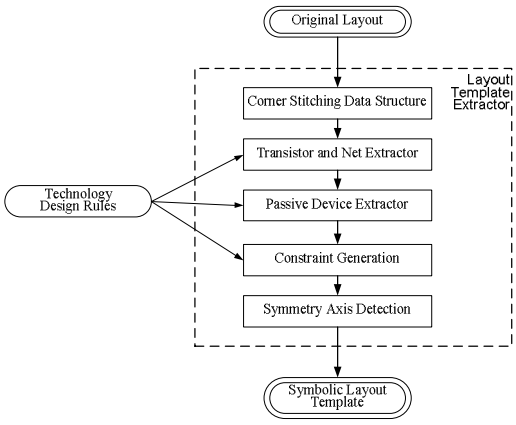

Fig. 2: Layout template extractor flow.

First, the original layout is stored in the *corner-stitching* data structure [7]. Here, each rectangle in the layout is stored explicitly as a *tile* and is linked to its neighboring tiles on lower-left and upper-right corners. Our preference for corner-stitching over other data structures, for example bins and linked-lists, is dictated by its efficiency in fast localized searches.

After the layout is stored, the MOSFET transistors are identified based on the overlap of multiple layers as defined in the technology design rules. Nets are then detected by searching for the connected neighboring tiles from all transistor terminal tiles in a depth-first-search manner. If vias or contacts are encountered, the search continues on different layers.

Specific layout patterns for passive devices are pre-defined in the technology design-rules. Passive devices, categorized into resistors, capacitors and inductors, are identified by net-traversing through the tiles and pattern-matching. Details of the passive device extraction and regeneration are discussed in section V.

As mentioned earlier, the original circuit topology, connectivity and matching have to be examined and reused in order to preserve design knowledge and integrities. Hence, the structural symbolic template has to possess an ability of maintaining the original layout's intellectual properties, an adaptability with new device sizes evaluated from new specifications, and a fast solvability. For these requirements, a *constraint graph*, which is widely used in the context of layout compaction [8], is selected. Here, each rectangular tile is represented by four independent nodes: left, right, top, and bottom tile edges. Constraints are placed between nodes in the graph to sustain layout integrity and correctness. These constraints are categorized into (1) connectivity, (2) design-rule, (3) symmetry, and (4) exact-device-size. Horizontal and vertical constraint graphs are constructed separately.

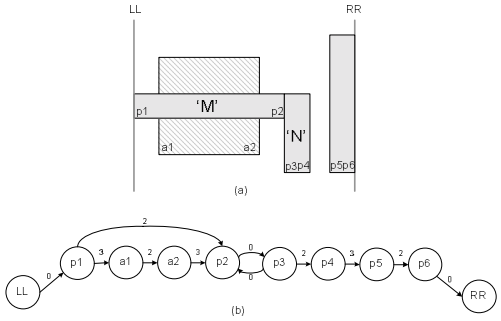

Fig. 3: Example of a constraint graph as symbolic template in horizontal direction.

Consider a sample layout of Fig. 3. The tile connectivity between *M* and *N* in the horizontal direction is retained by two constraint arcs of weight *'0'* between the edges *p2* and *p3*. The design rule constraints can be further sorted into three groups as minimum width (an arc from *p3* to *p4*), minimum spacing (an arc from *p4* to *p5*), and minimum extension (an arc from *a2* to *p2*). However, constraints due to symmetry, described in Fig. 4 such as the equal distance between *a1* to *a2* and *a3* to *a4*, cannot be directly added to the constraint graph. The handling of these constraints is explained in Section IIB.

Clearly, generating constraints from each node to every other nodes leads to significant redundancies. For example, adding a direct spacing constraint from *p2* to *p5* in Fig. 3 is redundant, because the minimum distance rule is already imposed by several arcs through *p3* and *p4*. To avoid this, a scan-line method [8] is employed.

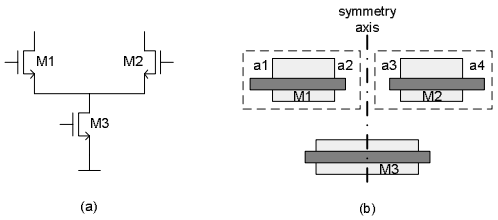

Fig. 4: An example of symmetry between transistors.

In the scan-line method, first, all rectangle edges are sorted by their abscissas. For each edge, constraint arcs are generated only from that edge to all other visible edges to its left, i.e. not blocked by other tiles. Progressing from the most-left to the most-right, the list of visible edges is updated when each scanned edge is completed. For example in Fig. 5, only arcs from *a* and *c* to *s* are included but not the one from *b* to *s*. The algorithm has a time complexity of $O(n^2)$, where *n* is the number of rectangle tiles.

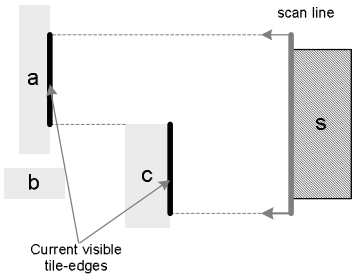

Fig. 5: Example of a scan line and its visible edges.

B. Layout Generation

After the structural symbolic template is constructed, different target layouts can be obtained by a two-step process: first, imposing new device sizes by modifying constraint arcs in the template, and second, solving the template through a combination of a linear programming and graph-based shortest-path algorithm. The detail procedures of the layout generator are shown in Fig. 6.

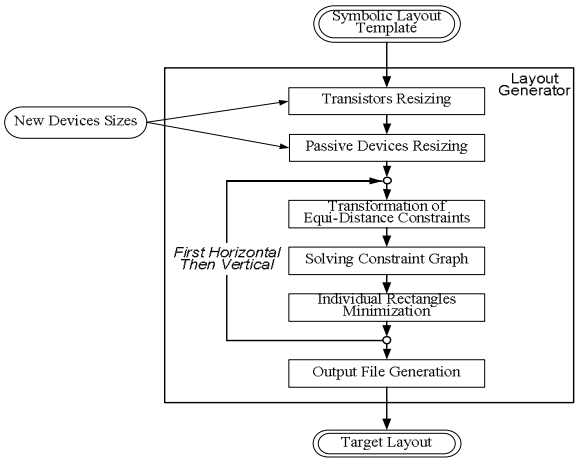

Fig. 6: Layout generator flow.

The new transistor sizes can be obtained from circuit simulations and optimizations performed manually by design engineers or automatically by circuit synthesis tools. Based on these, exact-device-size constraint arcs will be added to the graph symbolic template. For an example in Fig. 3, if the new transistor width is '*w*', an arc from *a1* to *a2* with weight '*w*' and an arc from *a2* to *a1* with weight '*(-1)×w*' will be added to the graph. Extra care in transistor resizing is focused on the active-metal contacts. If the size of a transistor is smaller in the target layout, the drain and source area may not be able to accommodate the original number of contacts. In this case, the template has to be updated so that the number of contacts can fit within the specific area. The algorithm for this is further explained in section IV.

Retargeting of the passive devices requires modification in device geometries and structures. A special scheme is implemented and is described in section V.

The assembled constraint graph template can be solved by applying the graph-based shortest-path algorithm on both horizontal and vertical templates separately [8]. The algorithm finds the shortest distance for every rectangle edge to the left (or bottom) layout boundary. This determines legitimate tile locations of the target layout, based on the design rules and the device sizes. The shortest-path algorithm has a worst-case time complexity of *O (nodes × arcs)*. However, symmetry information cannot be preserved by utilizing this algorithm only.

Prior to the graph solving, the symmetry constraints has to be converted into a form that can be imposed on the constraint graph [6]. At the beginning, a smaller *equivalent graph*, which consists of only nodes that appear in symmetry constraints, is created. The constraint arcs between nodes in the equivalent graph are present only when there is a direct path between a pair of nodes in the original graph template. Then, this equivalent graph is converted into a linear-programming equation form, in which graph-nodes are represented as variables and graph-constraint-arcs are represented as weight constrained equations between two variables. Together with the equations preserving the symmetry, the problem can be solved with integer linear programming, and the exact distance between each pair of variables can be computed. That distance is then imposed individually to the original graph template, thus the symmetrical distances are retained. For example in Fig. 4, if the distance between (*a1*,*a2*) and (*a3*,*a4*) is 'w_s', then four arcs – from *a1* to *a2* weight 'w_s', from *a2* to *a1* weight '$(-1)\times w_s$', from *a3* to *a4* weight 'w_s', and from *a4* to *a3* weight '$(-1)\times w_s$' – will be added. Afterward, the shortest-path algorithm can be carried out.

One weakness of the shortest-path algorithm is all edges are pulled toward most-left (or most-bottom), which creates excessive leftward extension in most rectangles. Therefore, an algorithm for minimizing individual rectangle areas as described in [9] has to be implemented to secure good layout.

III. Automatic Symmetry Detection

Since device symmetry in analog/RF layouts is critical to circuit performances [10], this information has to be

detected and maintained while generating target layouts. On a small layout, symmetries can be manually located easily. However, on a large and complex layout, especially on one containing transistors laid out in multi-finger fashion, an automatic approach has to be implemented.

First, a circuit netlist clustering all multi-finger transistors is extracted. By mapping it to known small-sized circuit netlists with easily identified symmetries, all the multi-finger transistor symmetrical pairs are located. After that, by sorting each unit transistor in the multi-finger group based on its relative position, each unit transistor symmetrical pair is detected. This symmetry information is then used to maintain the device matching, as explained in section II.

Fig. 7 shows an example of symmetric multi-finger transistors of *M1* and *M2*. The symmetry between *M1* and *M2* can be specified straightforwardly from the schematic diagram. Once the transistor-clustered netlist is created and mapped, *M1* and *M2* in the layout are declared symmetric.

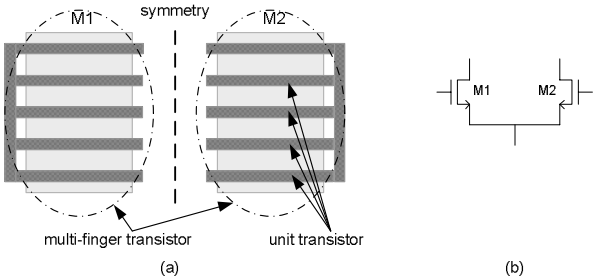

Fig. 7: Multi-finger transistor clustering for automatic symmetry detection.

IV. Via and Contact Removal

Due to high current density in analog and RF circuits, it is a common practice to layout connecting metal wires considerably wider than the design-rule minimum width. This practice also benefits the circuit performance as the parasitic effect of wire resistance and thermal noise is reduced [10].

Similarly, connections between two layers using vias or contacts also have to conduct large current. Therefore, high quality analog or RF layouts are frequently implemented with arrays of vias or contacts. Typically, one connection can contain 10's to 1000's vias or contacts.

This collection of vias or contacts exhibits two main challenges in the retargeting tool. First, it increases the size of the structural symbolic template, which results in longer template extraction time, longer layout generation time, and larger memory usage. Second, the presence of a fixed number of vias or contacts might limit the shrinkability of the tiles. Therefore, the following scheme is implemented to overcome these challenges.

During the layout representation in the corner-stitching data structure, each connected and overlapped rectangle-pairs in different layers are searched for the vias/contacts. The whole array of vias/contacts are then represented by only four rectangle tiles: top, bottom, left, and right, as shown in Fig. 8. Once the scan-line method is carried out and the structural symbolic template is created, two extra constraint arcs, *a* and *b*, are added to set the width and height of each array. Weights of those arcs are determined based on the number of vias/contacts on each connecting tiles and their related design rules. After solving the template for the target layout, vias/contacts in each connection have to be re-populated based on the available space.

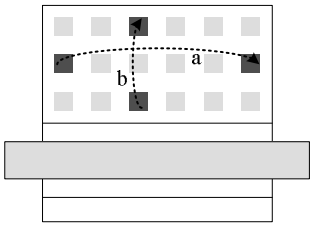

Fig. 8: Example of reduction in number of contacts. Only four contacts are kept for each rectangle pair.

The number of vias/contacts on the target layout can be adjusted based on the original rectangle sizes and aspect ratios, the original numbers of vias/contacts, or the new device sizes. Moreover, this will allow further sizing of connection metal wires based on parasitic effect or on current density.

V. Passive Device Retargeting

Along with transistors, passive devices – resistors, capacitors, and inductors – are also significant parts of analog/RF circuits. Retargeting those devices exhibits one critical issue as their geometries and rectangle structures might be modified. Merely updating rectangle widths and lengths are certainly not adequate. Here, a method of detection, representation in the template, and creation of new passive devices is presented.

Prior to the passive devices detection, the layout transistor netlist has to be extracted, in which all passive devices are embodied within the nets. Traversing through tiles in each net, a passive device is detected when the defined tiles structure is matched and the threshold geometry size is satisfied. Upon recognition, the traversed net is split, the device's geometry information is collected, and the port tiles are collected. A port is defined as a tile that connects a passive device to a corresponding net. The connection can occur on the same layer masks or on different layer masks connected through vias or contacts.

Examples of resistors and capacitors are shown in Fig. 9. On-chip resistors are generally implemented in poly-silicon layer as it exhibits high linearity and low capacitance to substrate [10]. In some technologies, n-well or active layer resistors are also used. The resistor topology supported consists of a single tile strip (Fig. 9a) or serially-connected multiple unit resistors (Fig. 9b). On-chip capacitors are commonly laid out in one of the two schemes. Firstly, a *P-P* (poly-silicon to poly-silicon) or *MIM* (metal insulator metal) capacitor (Fig. 9c) which is detected when two tiles of defined layers from two different nets overlap each other with adequate overlap area. Secondly, a MOSFET can also be used as a capacitor, referred as a MOSCAP, by shorting the source and drain terminals. Due to its structure, retargeting MOSCAPs is identical to transistors.

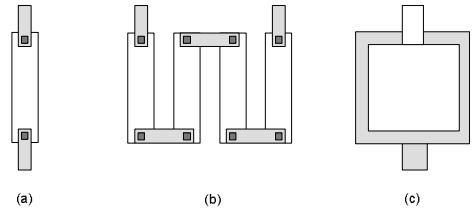

Fig. 9: Example of (a) single tile resistor. (b) multiple unit resistor. (c) MIM capacitor.

An inductor is commonly constructed with a planar spiral structure. The example of two-turns inductor is shown in Fig. 10a. This inductor structure is detected when a set of tiles of a defined inductor layer assembles at least one full turn. Limited by the corner-stitching data structure, only square shape inductors are supported.

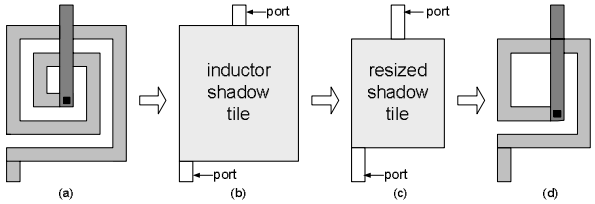

Fig. 10: Retargeting process of a planar square spiral inductor.

In the analog/RF layout, passive devices are usually isolated in space from other devices and nets to minimize the parasitic and coupling effects [10]. Therefore, to prevent an overlapping with other devices or net during the resizing process, a *shadow tile* is added on top of the passive device prior to the symbolic template generation. A shadow tile is a temporary non-physical-layer tile. It is used to allocate a dedicated area for the passive device by applying minimum spacing constraints to tiles on every layer.

Updating the passive device sizes is accomplished by adding exact-device-size constraints to the corresponding device tile variables for new sizes, and the shadow tile variables for reserving its dedicated space. In addition, more constraints are added between the shadow tile and the port tile variables in order to keep the passive device aligned and preserve the net connectivity after compaction.

Retargeting a more complex passive device structure, such as multi-turn inductor in Fig. 10a, requires a reconstruction of the tiles. First, in Fig. 10b, all device tiles, except for ports, are virtually removed from the layout and the entire device is represented by a single shadow tile in the symbolic template. During the layout generation process, the shadow tile area is adjusted according to the target device dimension, illustrated in Fig. 10c. Once the template is solved for a layout, the new passive device is independently regenerated, inserted into the allotted space, and re-connected to the corresponding ports, as shown in Fig. 10d. The new inductor is retargeted based on the geometry information, such as number of turns, outer and inner diameters, width, and spacing. These geometries can be obtained from a technology-based inductor library or through the inductor approximate expressions, for instance one proposed in [11].

VI. Experimental Result on Retargeting VCO Layouts

The automatic retargeting layout tool has been tested on RF circuits. The *CMOS complementary cross-coupled voltage controlled oscillator* was initially designed and laid out using the MIT Lincoln Laboratory's 0.18um low power FDSOI CMOS process. The VCO consists of 6 MOSFET transistors, 2 MOSCAP transistors, 2 output buffers, and 1 planar square spiral inductor. Each buffer is constructed of 3 transistors and one poly-silicon resistor. The schematic diagrams are shown in Fig. 11.

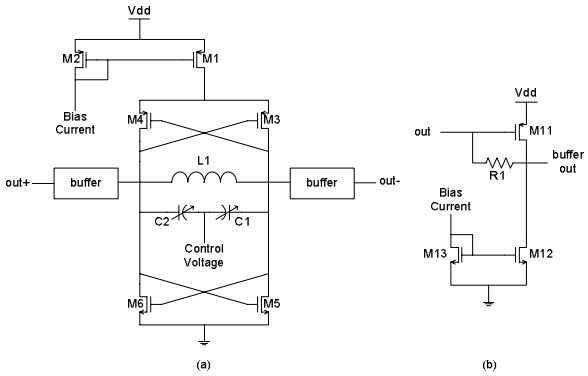

Fig. 11: Schematic diagram of (a) VCO and (b) buffer block.

The original layout is shown in Fig. 12. Each transistor is represented in multi-finger structures. The layout comprises of 202 unit transistors and 11 electrically connected nets.

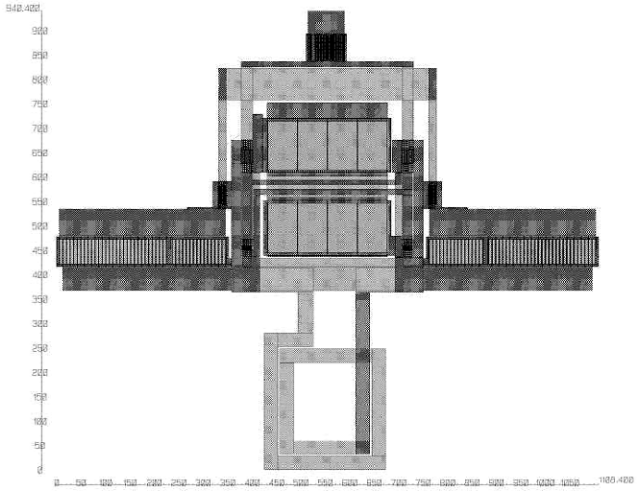

Fig. 12: Layout of an original VCO at 2.6GHz.

The tool was employed on the initial layout to generate a structural symbolic template. Originally, there were 310,673 rectangle tiles in the layout. After via/contact removal, the layout size was reduced to only 3,210 rectangle tiles. As for passive devices, two poly-silicon resistors and one planar spiral inductor were detected, and their shadow tiles were created. Since the target layouts would carry different inductor geometries, all the tiles constituting an inductor were removed. Then, the matching information between transistors was collected. The automatic symmetry detection based on netlist matching reported two

symmetrical axes with 79 pairs of symmetrical unit transistors from the layout. From Fig. 11, those were obtained from *M3*:*M4*, *M5*:*M6*, *C1*:*C2*, and *M11*, *M12* and *M13* between left and right buffers.

Finishing the extractor phase, the structural symbolic template, in a graph constraint form, consisted of 11,884 nodes, 205,878 arcs and 1,272 additional arcs extracted from symmetry constraints.

The template was utilized to generate three target layouts. Different sets of device sizes were manually designed based on different specifications. Next, transistor sizes and passive device geometries were enforced onto the template. Once the retargeting process was finished, the target layouts were design-rule checked (DRC), and their netlists were extracted and simulated in SpectreRF for functionality and performance verification. Each target layout specification, along with the initial layout specification, is listed in Table 1. Target layout III, which was designed for 5.6GHz oscillating frequency, is illustrated in Fig. 13.

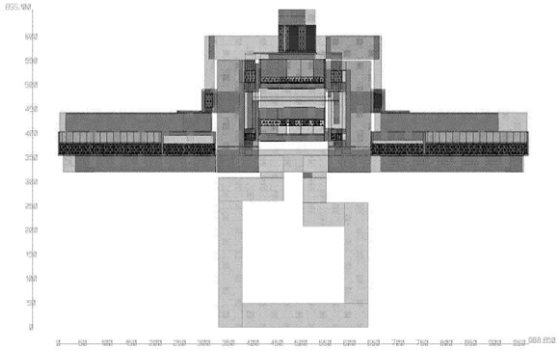

Fig. 13: Layout of a target VCO at 5.6GHz

Table 1. Specifications of original layout and target layouts

	Original Layout	Target Layout I	Target Layout II	Target Layout III
Oscillating Freq.	2.6 GHz	3.8 GHz	4.5 GHz	5.6 GHz
Phase Noise at 60kHz	-98 dBc/Hz	-96 dBc/Hz	-103 dBc/Hz	-98 dBc/Hz
Tuning Range	6 %	5 %	7 %	4 %
Output Swing	2.0 V	2.0 V	2.2 V	2.4 V
Inductor	1.0 nH	0.5 nH	0.5 nH	0.4 nH
Power	7.4 mW	9.5 mW	5.2 mW	7.0 mW
Area	1.04 mm^2	0.74 mm^2	0.71 mm^2	0.63 mm^2

The automatic retargeting program was run on a 900MHz SUN UltraSparc3 workstation. The layout template extractor phase took 1 hour 48 minutes. Once the template was created, the generation of target layout I was completed in 7.56 minutes, target layout II in 8.43 minutes, and target layout III in 9.19 minutes.

VII. Summary

In this paper, a tool capable of retargeting analog/RF layouts to different specifications is presented. A structural symbolic template is automatically generated so that the analog/RF layout integrity is preserved. New transistor sizes and passive device geometries, obtained from various specifications, can be imposed onto the template to generate several target layouts. The tool was applied successfully to produce three different operational VCO layouts within minutes after the template had been extracted.

Despite the restriction in circuit topology changes due to template extraction, the retargeting tool exhibits several key benefits. Its speed, its template reusability, and the fact that it does not require much designer involvement make this a great potential tool for many analog and RF circuit retargeting applications.

References

[1] M. D. M. Hershenson, A. Hajimiri, S. S. Mohan, S. P. Boyd, and T. H. Lee, "Design and optimization of LC oscillators", *Proc. of IEEE/ACM Int. Conference on CAD*, pp. 65-69, November 1999.

[2] M. Aktuna, R. A. Rutenbar, and L. R. Carley, "Device-level early floorplanning algorithms for RF circuits", *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 18, pp. 375-388, April 1999.

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

[4] R. Castro-Lopez, F. V. Fernandez, F. Medeiro and A. Rodriguez-Vazquez, "Generation of technology-independent retargetable analog blocks", *Int. Journal of Analog IC and Signal Processing*, Kluwer Academic Publishers, vol. 33, pp. 157-170, December 2002.

[5] N. Jangkrajarng, S. Bhattacharya, R. Hartono, and C-J. R. Shi, "Automatic analog layout retargeting for new processes and device sizes", *Proc. of IEEE Int. Symposium on Circuits and Systems*, vol.4, pp. 704 -707, May 2003.

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

[7] J. K. Ousterhout, "Corner stitching: a data-structuring technique for VLSI layout tools", *IEEE Trans. on CAD of Integrated Circuits and Systems*, pp. 87-100, January 1984.

[8] S. L. Lin and J. Allen, "Minplex – a compactor that minimizes the bounding rectangle and individual rectangles in a layout", *Proc. of IEEE/ACM Design Automation Conference*, pp. 123-130, June 1986.

[9] G. Lakhani and R. Varadarajan, "A wire-length minimization algorithm for circuit layout compaction", *Proc. of IEEE Int. Symposium on Circuits and Systems*, pp. 276-279, May 1987.

[10] B. Razavi, *Design of analog CMOS integrated circuits*, McGraw Hill, 2001.

[11] S. S. Mohan, M. D. M. Hershenson, S. P. Boyd, and T. H. Lee, "Simple accurate expressions for planar spiral inductances", *IEEE Journal of Solid State Circuits*, vol. 34, pp. 1419-1424, October 1999.