LAYGEN II – Automatic Analog ICs Layout Generator based on a Template Approach

Ricardo Martins
Instituto de Telecomunicações
Instituto Superior Técnico
IST–Torre Norte, AV. Rovisco Pais
1049-001 Lisboa, Portugal
ricmartins@lx.it.pt

Nuno Lourenço
Instituto de Telecomunicações
Instituto Superior Técnico
IST–Torre Norte, AV. Rovisco Pais
1049-001 Lisboa, Portugal
nlourenco@lx.it.pt

Nuno Horta
Instituto de Telecomunicações
Instituto Superior Técnico
IST–Torre Norte, AV. Rovisco Pais
1049-001 Lisboa, Portugal
nuno.horta@lx.it.pt

ABSTRACT
This paper describes an innovative analog IC layout generation tool, LAYGEN II, based on evolutionary computation techniques. The designer provides the high level layout guidelines through an abstract layout template. The template contains placement and routing constrains independently from technology, and can be used hierarchically in the definition of templates for complex circuits. LAYGEN II uses this expert knowledge to guide the evolutionary optimization kernels during the automatic layout generation in the target technology. The routing task of the proceeding can range from a template-based approach to a full automatic generation, if only connectivity is provided. The LAYGEN II tool is demonstrated for the layout generation of two typical analog circuit structures and the results validated by Calibre® design rule check tool.

Categories and Subject Descriptors
I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems – *industrial automation.*

General Terms
Algorithms, Performance, Design, Reliability, Verification.

Keywords
Integrated Circuits, Evolutionary Computations, Electronic Design Automation, Computer Aided Design, Physical Design

1. INTRODUCTION
In the last few years, the world has observed the increasing complexity of integrated circuits (ICs), strongly triggered by the proliferation of consumer electronics. Thanks to the developments made in the last decades in the area of very large scale integration technologies, designers have the means to build multimillion transistor ICs, meeting the needs of an ever-increasing microelectronics market. The design of complex systems-on-a-chip (SoCs) is emerging in telecommunications and multimedia applications, these systems merge on the same chip analog or mixed-signal blocks together with digital processors and memory blocks [10]. Moreover, it is known that most functions in today's ICs are implemented using digital signal processing circuitry. Analog blocks only constitute a small fraction of the components on mixed-signal ICs and SoC designs, being, essentially, the link between digital circuitry and the continuous-valued external world, so are also integrated on the same die [9].

GECCO'12, July 7–11, 2012, Philadelphia, Pennsylvania, USA.

However, the development time of analog blocks is much higher when compared to the development time of the digital blocks. Analog circuits are known for its difficult re-utilization, so designers have been replacing analog processing for digital computations, but there are some typical blocks in today's ICs appointed as remaining analog forever. The two main reasons for the larger development cycle of analog blocks identified are the lack of effective computer-aided-design (CAD) tools for electronic design automation (EDA), since analog design is less systematic, more knowledge-intensive and more heuristic in nature than digital counterpart; and that analog circuits are being integrated using technologies optimized for digital circuits. For these reasons, given the rampant growth of analog and mixed-signal (AMS) systems, the economic pressure for high-quality yet cheap electronic products and time-to-market constraints, there is an urgent need for CAD tools that increase the analog design productivity and improve the quality of resulting ICs [31].

The analog IC design requires designer knowledge and circuit design skills acquired through many years of experience, even at low-level, unlike the digital domain where EDA is well developed and establish an almost fully automated low-level design process. Today's analog design is supported by circuit simulators, layout editing environments and verification tools, which maintain the design cycle for AMS ICs long and error-prone. These circuits suffer from diverse non-idealities and parasitic disturbances that, by not being weighted in the early stages of development, can be responsible for design errors and expensive re-design cycles, becoming the bottleneck of SoC and mixed-signal ICs design. Therefore, analog level of automation is far from the "push-button" stage.

This paper describes a methodology for automatic analog ICs layout generation, which is an evolution from the previous LAYGEN [23] approach. The template captures the designer knowledge independently of technology, through introduction of an abstraction level between technological details and guidelines. This design approach focus on improving design reusability and retargetability once the template is available. It introduces a new level of flexibility by supporting changes on device and modules specifications, and different levels of automation in the synthesis process. In addition, the designer may opt for some tasks of the proceeding which can range from a template-based approach to an automatic generation, allowing the tool to explore the design space. Design productivity is increased only if the target layout can be automatically generated in a process guided by the designer, and the result validated with a commercial tool, assuring the quality of the solution.

This paper is organized as follows: In section 2, a general overview on typical layout generation procedures is given. Afterwards, in section 3, the general architecture of LAYGEN II is described. In section 4, some case studies are presented. Finally, in section 5, the conclusions are addressed.

2. PREVIOUS WORK

Before presenting state of the art approaches in analog layout generation, a brief introduction on analog design flow is provided.

2.1 Analog Design Flow

Different analog design flows are available in literature, however the majority of the works developed in the last decade follow the design flow introduced by Gielen and Rutenbar [9]. This design flow for AMS ICs, illustrated in Figure 1, consists of a series of top-down design steps repeated from the system level to the device-level and bottom-up validation. The number of hierarchy levels depends on the complexity of the system being handled, and the steps between any two hierarchical levels are:

- Top-down electrical synthesis path includes topology selection, specification translation (or circuit sizing at lowest level) and design verification;
- Bottom-up physical synthesis path includes layout generation and detailed design verification (after extraction).

Layout generation consists of generating the geometrical layers for a given technology of the circuit block under design at the lowest level in the design hierarchy, or place and routing the layouts of the sub-blocks at higher levels. The presence of a detailed verification step that acts over the extraction of the layout is required in order to ascend to higher hierarchy levels, only if the layout meets target requirements. When the top-most level verification is complete, the system is designed. LAYGEN II focuses on the layout generation of analog circuits, appointed as the critical part of the analog design flow [9][11]. Although it works as a standalone tool, it is being developed to be integrated in the bottom-up physical synthesis path of a design automation process. Namely, LAYGEN II receives an optimal sized circuit performed by an in-house automatic analog IC sizing tool, GENOM-POF [3, 5][24].

2.2 State of the Art

Having the devices for the selected topology sized, they must be laid out in the chip. A common approach is to split layout generation in two smaller problems, placement and routing. Placement refers to the assignment of the device locations in the chip, and routing to the interconnections between those devices.Each placement tool has its own strategy of representing the cells' location. To reduce unwanted impact of process variations and improve the circuit performance, topological constraints like device matching, symmetry and proximity have to be considered into the placement task. Absolute approaches represent cells by means of absolute coordinates. Topological representations encode the positioning relations between any pair of cells. The optimizer does not move cells explicitly, instead, it alters the relative positions of cells by modifying the structure encoding the layout [11]. There are two classes of topological representations: slicing and non-slicing. Since not all the layout topologies have a slicing structure, the first representation can degrade the density of the placement solutions, which is aggravated because cells in analog circuits are usually very different in aspect ratio.Several non-slicing structures have been presented in the past. Sequence pair (SP) encodes the "left-right" and "up-down" positioning relations between cells [1], while the bounded-sliceline grid (BSG) [28] uses a meta-grid structure without physical dimensions, introducing orthogonal relations of "right-of" and "above". The ordered tree [12] extended the binary tree to the representation of non-slicing structures, and an efficient upgraded representation of binary trees (B*-tree) [7] is also available. More recently, the concept of Symmetry Island [20] was introduced, which keeps modules of the same symmetry group connected to each other. A transitive closure graph-based (TCG) [14] and similar approaches were proposed to combine the advantages of SP, BSG and B*-tree representations.

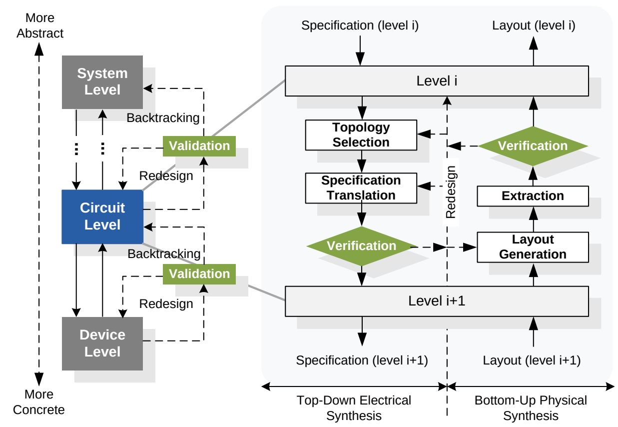

Figure 1. Hierarchical level and design tasks of analog design flow architecture.

Over the years different placement tools have explored the advantages of several chip floorplan representations and new ways of treating layout constraints, these approaches had been integrated with more or less success in analog synthesis tools. A method of symmetric placement by linear programming can be found in [18], while an algorithm based on hierarchical module clustering was presented by Lin et al. [21]. A full deterministic approach can be found in Plantage [32], which is based on a hierarchically bounded enumeration of basic building blocks. In a different direction a thermal-driven analog placement solution was proposed [22].

The earliest approaches for analog layout generation, procedural module generation techniques coded the entire layout of circuit in a software tool, which would generate the target layout for the parameters attained during sizing. ALSYN [27] employs fast procedural algorithms that are controlled through a database structures and attributes. A high-functionality pCell library independent of technologies can be found in [16]. Due to the fast processing of basic cells, procedural-based layout generation was used during sizing task by Vancorenland et al. [34] and Ranjan et al. [29]. Castro-Lopez et al. [6] also uses parametric layout generators during automatic sizing, enabling the awareness of parasitic and geometric variables, such as area or aspect ratio. Although fast processing, these methods may lack of flexibility, since the cost for introducing a new design task is relative high and technology migrations may force complete cells redesign.

A template-based generation is used by IPRAIL (Intellectual Property Reuse-based Analog IC Layout) [15] to automatically extract the knowledge embedded in an already made layout, and use it for retargeting. Layout retargeting is the process of generating a layout from an existing one, the main target is to conserve most of the design choices and knowledge of the source design, while migrating it another given technology, update specifications or attempt to optimize the old design [11]. In order to retain the knowledge of the designer, but without forcing an implicit definition of the layout, LAYGEN [23] uses a template to guide the layout generation. ALADIN [36] also allow designers to integrate their knowledge into the synthesis process, while in ALG [35] the designer can interact with the tool in different phases. Zhang et al. [37] developed a tool that that automatically conducts performance-constrained parasitic-aware retargeting and optimization of analog layouts.

The fully automatic place-and-route layout generation approaches consist of synthesizing the layout solution using optimization techniques with a higher level of abstraction. In ILAC [30] a slicing structure representation is used to limit the search space. However, representing the cells by means of absolute coordinates proved to be the most practical solution to implement layout constraints, even though it allows for an infinitely large solution space. This is the approach found in KOAN/ANALGRAM II [8], LAYLA [19], Malavasi et al. [25] and ALDAC [17]. Recently, Habal et al. [13] ruled out the use of procedural generators during sizing task, investigating every possible layout for each device using Plantage [32] algorithm. These methods are usually slow and not always produce optimal solutions in terms of area and performance.

Figure 2 establishes a chronological representation of the tools presented in this section, organized by the placement generation technique used.

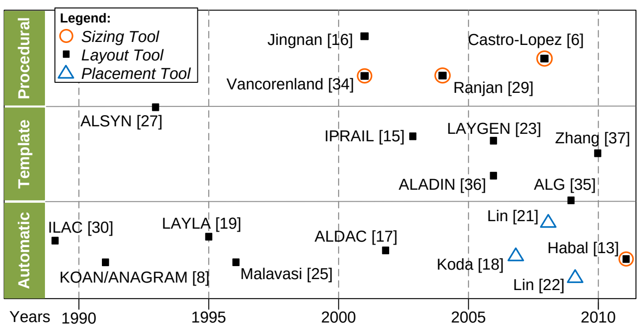

Figure 2. Chronological representation of analog layout design tools.

3. ARCHITECTURE

It is acknowledged that each designer/company has its own layout style but often this style is very regular. For a large number of applications, even with some specifications or technological changes, the design guidelines for most common cells are kept the same. For simple cells, parametric generators are a valid solution to implement these guidelines. However, though technological detail may also be included as parameters, these module generators are highly dependent of technology making them difficult to reuse. In addition, for complex cells the development of effective parametric generators has proven ineffective, either on design-time or design-reusability.

In order to cope with these limitations, our approach stores these design regularities in a layout meta-description that is independent of technology. The template, together with LAYGEN II and a set module generators at device-level, provide the designer with a technological and specification independent way of defining some of the most commonly used cells.

Figure 3 depicts principal tasks performed by LAYGEN II, with emphasis on the two optimization kernels used. The output provided is a GDSII stream format, a file standard in the microelectronics industry. The physical validation of the result is performed in Mentor Graphics' Calibre® [26] Design Rule Check (DRC) tool, a main reference in the ICs design when the development is intended for fabrication.

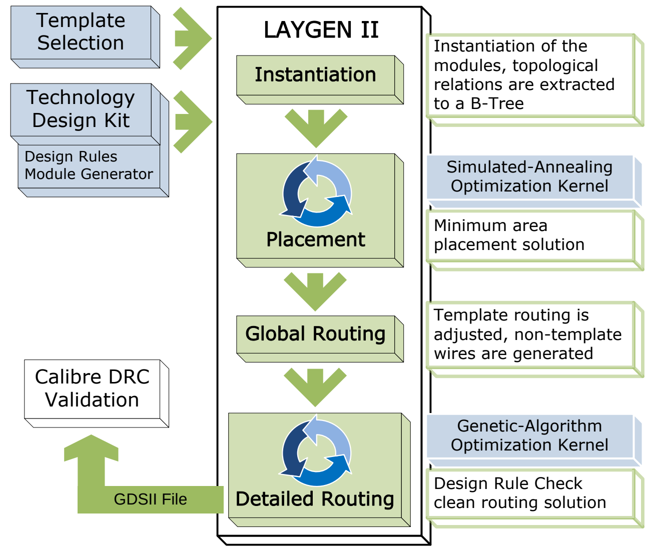

Figure 3. LAYGEN II Architecture.

In the next sub-sections some detail will be provided for the principal tasks performed by LAYGEN II.

3.1 Hierarchical High Level Cell Description

As mentioned, to reduce unwanted impact of parasitic and process variations, many knowledge-intensive constraints have been considered in analog design. Device matching, symmetry and proximity, current density in interconnects and thermal effects are just some of the factors that analog designers have to consider while planning an analog layout project. While designer's expertise is essential in this phase, this type of knowledge does not require being aware of the particular details of a given technology, e.g., minimum distances and enclosures allowed between layers.

The designer expertise is caught into the technology independent template and used to guide the automatic layout generation. These guidelines will increase the layout quality, reducing the solution space and thus the computational efforts required to achieve the pretended solution.

The template information used for placement is device sizes, expected relative placement of the devices, symmetry and matching requirements. Also, each device is attached with a set of different, but electrically equivalent, layout representations, henceforward addressed as modules. Templates can also be used as modules in a hierarchically manner, allowing the designer to use templates for simplier cells in the definition of more complex ones, splitting the complexity of the space search into different executions of the optimization kernels.

For routing, the designer may define relative position and geometry of each net, ensuring that the pretended shape for a given wire is present in the final layout. If only connectivity is provided, henceforward addressed as non-template wires, the tool is free to explore different valid solutions. Each template may be defined for any range of non-template wires, being at the discretion of the designer the automation level of routing.

Figure 4 presents a cascode current mirror and a possible template for his layout generation. A routing space is defined between each pair of transistors to avoid routing considerations during placement. Only wires shapes connecting transistor gates were defined by the designer, being the remainder automatically generated in a process detailed in following section 3.3.

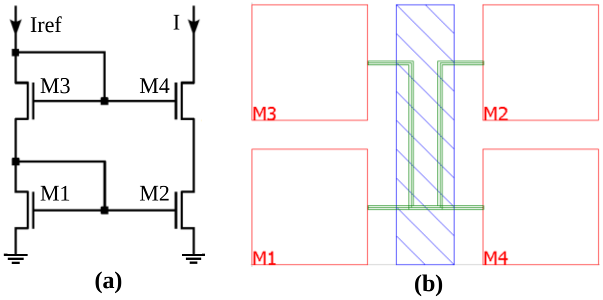

Figure 4. (a) Cascode current mirror. (b) Template view.

With this template description, designer has a way to control the process at a higher level, being the LAYGEN II responsible for dealing with the exact placement and routing, while attending the specific design rules to the target technology and the device sizes specific to the target application. The layout generation proceeds in the traditional way, first placement and then routing.

3.2 Placer

The topological relations present in the template are mapped to a non-slicing B-Tree layout representation, on which the O(n log n) packing algorithm presented in [2] is used to obtain a compact placement. The correspondence between an admissible placement and its B*-tree is one-to-one, so no redundancy.

To decide the optimal combination of modules an evolutionary Simulated-Annealing-based (SA) [33] optimization kernel is used. The placer explores the alternative placements by selecting one of the alternative modules for each to device and packing the layout. Since this process is a macro cell placer without overlap, the optimizer objective is the minimization of the effective area occupied, restrained to the topological relations present in the template. If desired, designer can restrict the obtained placement to a certain aspect ratio, which can be useful for the sub-templates in a bottom-up hierarchical template description approach.

When the final B*-tree packing is obtained, the modules are placed in the floorplan at the minimum distances allowed by target technology.

3.3 Router

The router uses the placement solution and the template's information to produce the desired routing. The major challenge in the router is the design rules verifications; they make the routing algorithm extremely complex and computationally more expensive than the placement algorithm.

Each net is divided into a set of wires, each one connecting two and only two contact points (pins). Each wire is formed by any number of linked line-segments. For its part, a line-segment refers to the connection of two different points in the solution space, whose values in the x axis or in the y axis are the same.

The routing algorithm used is a two-step procedure. First global routing, which coincides with the generation of the initial population of the genetic optimizer, and then the optimization kernel attempts to improve the detailed routing quality.

3.3.1 Global Routing

First, the template routing is adjusted (re-scaled) to the newly created placement. This operation is required because the wires geometries were defined pre-placement, abstractly from the real pins locations in the target layout. The adjustment procedure encompass the following: the template wires are scaled, then, moved to set the wire start point on the new start-pin's position, and finally, the wire end position is set to the new end-pin's position to ensure connectivity.

Moreover, the non-template wires are randomly generated through a set of heuristics; the possible geometries are presented in Figure 5. It is important to notice that in heuristics (d)-(g), the line-segments inside the wire structure, that is, those not connected to source pin neither sink pin, can take any position in the range of x axis or y axis allowed by the distance between terminal pins, sharply increasing the solution space. Since the only restriction is the effective pins location, the optimizer has to ensure connectivity and technology design rules validation, without preserving a designer's desired shape.

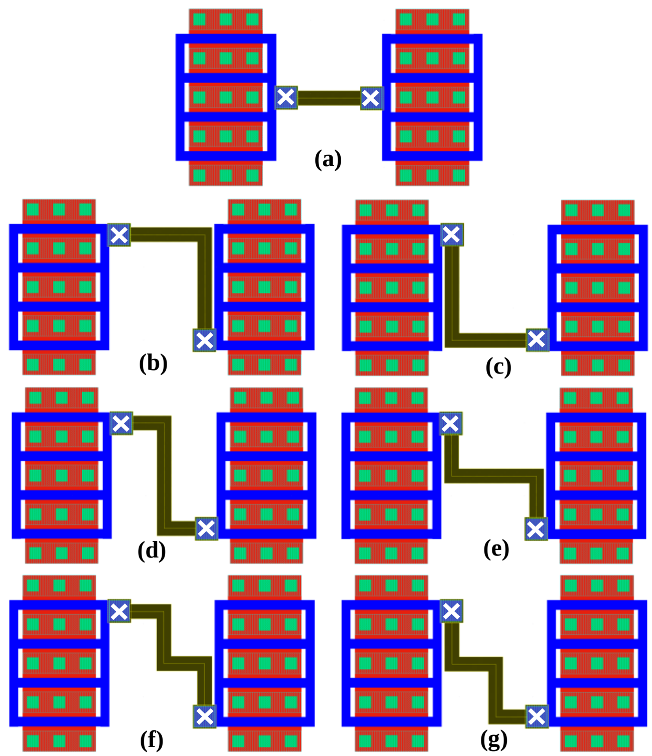

Figure 5. Different heuristics for generation of wires with (a) one, (b)(c) two, (d)(e) three and (f)(g) four line-segments.

3.3.2 Genetic Optimizer

Global routing is used as the start point for the evolutionary optimizer. Each element in the population, chromosome, encodes the information of a different routing solution, corresponding each gene to one single wire. So, each chromosome has a number of genes equal to the number of wires required to accomplish the routing. For its part, each gene has a variable size dependable on number of line-segments used to define a wire; still, every line-segment has its own associated layer.

In Figure 6 (a), an example layout for the cascade current mirror circuit of Figure 4 is presented, and also, two different representations of the chromosome for the current routing, Figure 6 (b) and (c). For this abstract technology two different levels of conductor layers for routing, pin 'X' denote a channel (via) between different conductors, were considered.

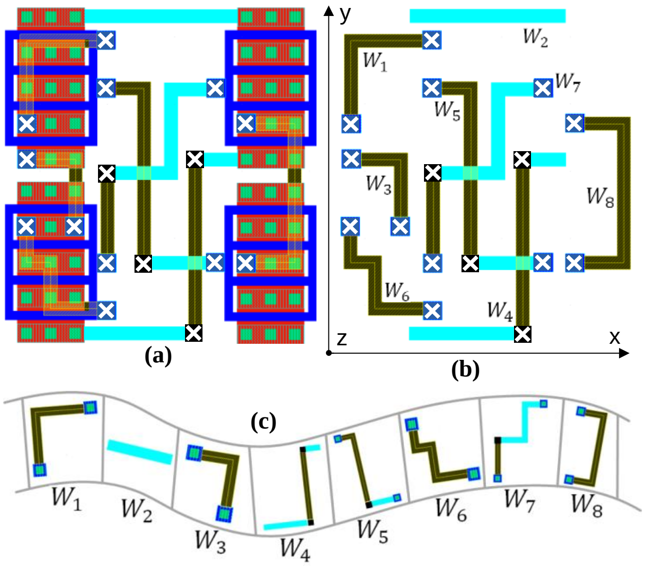

Figure 6. (a) Layout for the cascade current mirror. (b) Physical representation of the chromosome. (c) Abstract representation of the chromosome.

For the provided example although only 8 wires are necessary, they represent the complexity of problem for optimization. For each different layer, one shape of a given wire can't intersect another shape from a different wire, and still has to respect a minimum distance imposed by technology. Due to the countless interactions between shapes, layout solution is very restricted. This solution space grows rampantly not only with the number of wires necessary for a given routing, but also with number of possible different heuristics considered for their structure.

The evaluation of routing at each generation is performed by powerful internal DRC functions, which verify if each geometry layer in the layout meets the technology designs rules. Verifications must be made not only between wires, but also considering device's layers. This methodology follows a correct-by-construction generation, this is, LAYGEN II has the methods required to predict if a layout will be successfully validated.

The results of this internal DRC are the constraints of the optimizer. Namely, short circuits, when a wire crosses another different wire or device in the same layer; and minimum distances violations. Design rules of enclosure and extension between layers were previously considered in the module generator. The elements of population are ranked at each generation, benefiting those who present fewer errors. When those constraints are fulfilled, the optimizer minimizes a designer defined fitness function, that can incorporate different objectives like total wire length, minimum distance between nets to separate them as far as possible reducing crosstalk, number of conductor levels used, etc.

While non-template wires cause a great diversity within the genes of a population, one template wire is basically a common gene to each chromosome of the population. This common gene although it may have different physical representations, it has the same number of line-segments and structure, subsisting this way through the genetic operators to the final layout.

3.3.2.1 Crossover

At each new generation, each pair of parents is selected by tournament to generate two offspring that present a combination of their wires. A multi-point crossover is used as presented in Figure 7.

P1	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8
P2	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8
O1	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8
O2	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8

Figure 7. Example of crossover, parents and offspring.

When a gene of both parents present the same number of line-segments and structure, the crossover operator goes further than just select the wire of one parent to form the offspring, as for example marked in Figure 7, in the wire W_4. To increase the solution search, the offspring wire can be a combination of the two parents' wires, as depicted in Figure 8. The parents' contribution refers not only to wire shape, but also the combination of layers used. The same analogy can be established for wires with higher number of line-segments, always defining offspring solution space in the area equal and between the structures of equivalent line-segments of the parents.

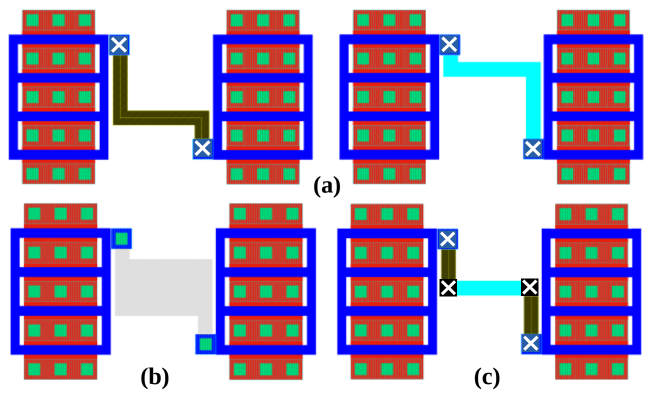

Figure 8. Crossover of two equivalent parents: (a) Parents. (b) Solution space for the offspring. (c) Possible offspring.

3.3.2.2 Mutation

A mutation ratio is applied to each of the chromosomes of the offspring population. There is a set of operators to be applied to the wires; those will introduce diversity for the further evaluations. The geometric operations applied are the following:

- Slide one line-segment inside the wire structure (Figure 9 (a));
- Slide two line-segments inside the wire structure (Figure 9 (b));
- Remove one line-segment from the wire structure (as extreme sliding consequence);
- Recover a line-segment previously removed from the wire;
- Randomize wire shape (affects only non-template wires).

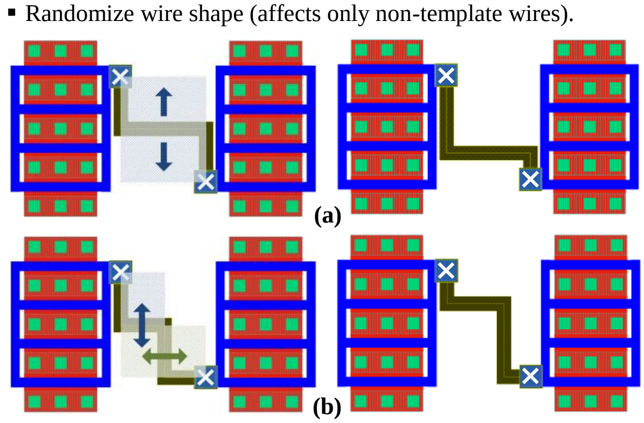

Figure 9. Slide (a) one and (b) two line-segments operator.

Since template wires have a pre-defined shape, the slide operators are essential to introduce some diversity. With these operators the lowest levels of conductor layers are used more effectively, which otherwise, most of the routing problems could easily be unfeasible. The layer shifting operations applied are:

- Move one line-segment in the wire structure to the conductor layer immediately above;
- Move one line-segment in the wire structure to the conductor layer immediately bellow;
- Move one line-segment in the wire structure to a randomly selected layer.

The presented operators were implemented in the evolutionary kernel and the results obtained are remitted to the next section.

4. IMPLEMENTATION AND RESULTS

The proposed design methodology has been coded in JAVA and is running, for the presented examples, on an Intel® Core™ 2 Quad CPU 2.4 GHz with 6 GB of RAM. The code automatically generates the GDSII file required by Calibre® DRC tool. Results are presented in LAYGEN II graphic user interface (GUI) for the two addressed examples. Layouts were generated for a 0.35µm CMOS design process and 4 different levels of conductor layers were used for routing.

4.1 Two Stage Single Ended Amplifier

The layout of Figure 12 was automatically generated for the amplifier of Figure 10, using the hierarchical template of Figure 11 and the device sizes presented in Table 1, which were obtained by a prior sizing task.

Table 1. Parameters attained during sizing task.

Device	Size		Partition
	Width	Length	
M0/M1/M5/M7/M10/M11	1,5 µm	10,0 µm	1
M6/M8	20,0 µm	1,0 µm	1
M2/M3	50,0 µm	0,75 µm	2
M4	100,0 µm	0,75 µm	2
M9	80,0 µm	0,75 µm	2
C1/C2	40,0 µm	32,0 µm	Top

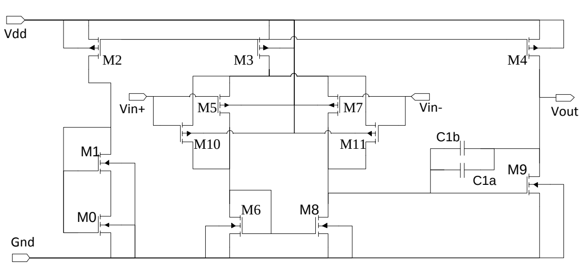

Figure 10. Circuit schematic.

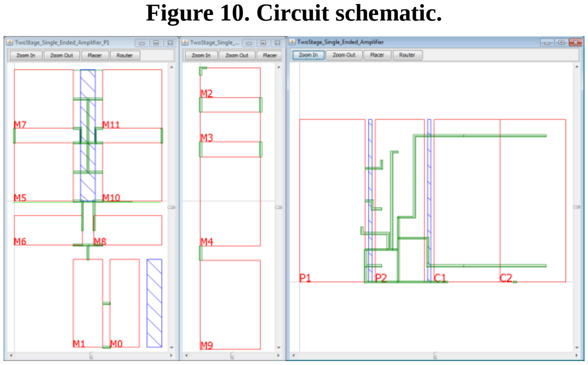

Figure 11. Hierarchical template description. (GUI)

Results were generated for a population of 64 elements, 300 generations and a mutation rate of 30%. Cell hierarchy was generated in 174 seconds, where routing optimization dominated more than 99% of the computational time, and the results were successfully validated by Calibre® DRC tool.

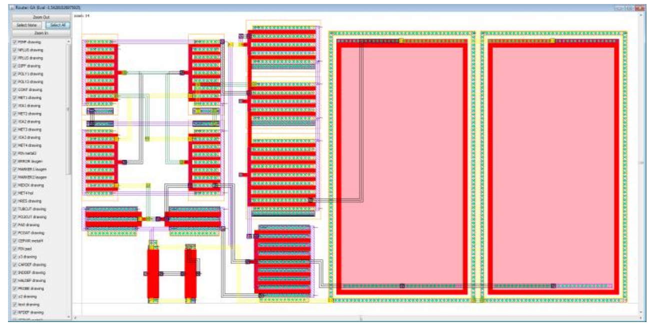

Figure 12. Automatically generated layout. (GUI)

In order to test the performance of the genetic evolutionary kernel, devoted to routing, all template wires from sub-partitions were moved to the top cell, increasing the complexity of the problem. In this case the placement was performed at sub-partitions. Next, the convergence of the algorithm, measured in terms of DRC violations, is tested using different optimization parameters. In Figure 13, although only 64 and 128 element populations have converged to zero DRC violations in the timeline of 300 generations, the computation times are far higher when compared to small populations. It is legit to use small populations with higher generation periods, to obtain equally DRC clean solutions. For a template-based routing, higher mutation rates proved to disperse the offspring population as depicted in Figure 14.

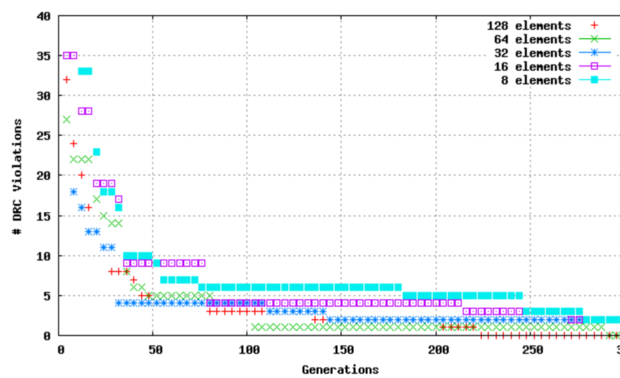

Figure 13. Evolution of DRC violations with the number of generations, with a 30% mutation ratio.

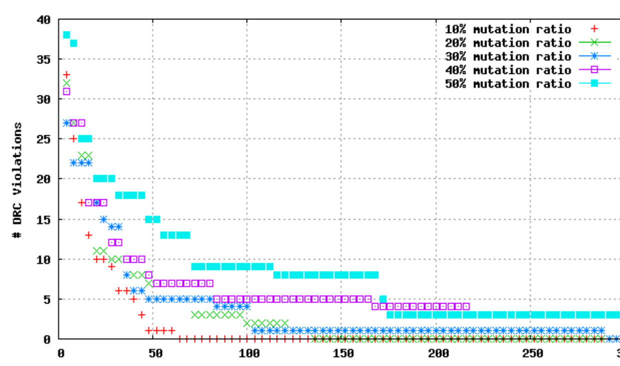

Figure 14. Evolution of DRC violations with the number of generations, with 64 elements in population.

4.2 Two Stage Cascode OTA

The layout of Figure 16 was automatically generated for the operational transconductance amplifier (OTA) of Figure 15, to exploit the potential of the automatic routing, since only connectivity was provided in the hierarchical template description. The above results were generated for a population of 64 elements, 500 generations and a mutation ratio of 20%. The computational time for this problem was about 202 seconds. The generation of a full automatic routing is accomplished in reasonable computational times. Although, the automatic generation depends on the circuit complexity, this problem is here mitigated by using hierarchical and modular descriptions.

Table 2. Parameters attained during sizing task.

Device	Size		Partition
	Width	Length	
M21/M22	60,0 µm	0,35 µm	1
M3/M4	160,0 µm	0,9 µm	1
M5A/M5B	20,0 µm	0,6 µm	1
M11/M12	60,0 µm	4,0 µm	2
M23/M24	54,0 µm	3,0 µm	2
M3A/M4A	60,0 µm	0,45 µm	2
M1/M2	120,0 µm	0,9 µm	3
M1A/M2A	100,0 µm	0,35 µm	3
C1/C2	50,65 µm	25,8 µm	Top

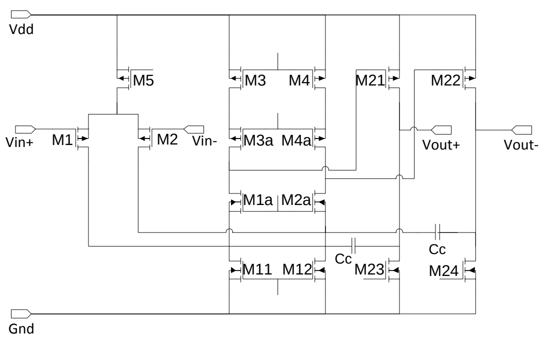

Figure 15. Circuit schematic.

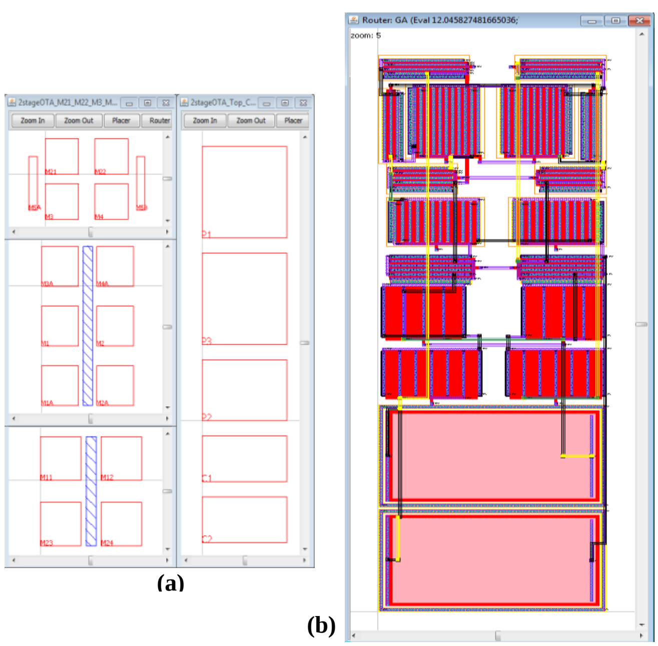

Figure 16. (a) Hierarchical Template Description. (b) Automatically generated layout. (GUI)

5. CONCLUSIONS

The proposed methodology for the automatic generation of analog IC layouts was proved by the implementation of a tool, LAYGEN II, which is able to generate robust layout solutions validated by a commercial tool widely accepted by the industry, Calibre® DRC. The proposed approach is template-based allowing the designer to provide layout guidelines, which is used as a first cut solution allowing an intelligent pruning of the design space and, therefore, reducing the overall computational effort required by the evolutionary optimization kernel. Moreover, the use of a technology independent template easies the migration of designs to different IC technologies. The hierarchical and modular nature of the developed approach allows the generation of large circuits layout by scaling the problem into different sub-templates. Additionally, the introduction of automatic generation options during routing, allows the designer to explore different layout topologies without the effort of defining new templates. Finally, the tool was validated for a wide range of real IC design cases from which two were presented.

6. ACKNOWLEDGMENTS

This work was partially supported by the Fundação para a Ciência e Tecnologia (Grant FCT-DFRH-SFRH/BD/72698/2010) and by the Instituto de Telecomunicações (Research project AIDA - IT/LA/1112/2011).

7. REFERENCES

[1] Balasa, F., and Lampaert, K. 2000. Symmetry within the sequence-pair representation in the context of placement for analog design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.*19, 7 (Jul. 2000), 721-731.

[2] Balasa, F., Maruvada, S.C., and Krishnamoorthy, K. 2003. Using Red-Black Interval Trees in Device-Level Analog Placement with Symmetry Constraints. In *Proceedings of the Asia and South Pacific Design Automation Conference* (Jan. 2003), 777-782.

[3] Barros, M., Guilherme, J., and Horta, N. 2007. GA-SVM feasibility model and optimization kernel applied to analog IC design automation. In *Proceedings of the 17th ACM Great Lakes symposium on VLSI* (Mar. 2007), 469-472.

[4] Barros, M., Guilherme, J., and Horta, N. 2010. Analog circuits optimization based on evolutionary computation techniques. *Integration, the VLSI Journal*. 43, 1 (Jan. 2010), 136-155.

[5] Barros, M., Guilherme, J., and Horta, N. 2010. *Analog circuits and systems optimization based on evolutionary computation techniques*. Studies in computational intelligence 294, Springer.

[6] Castro-Lopez, R., Guerra, O., Roca, E., and Fernandez, F. 2008. An integrated layout-synthesis approach for analog ICs. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.* 27, 7 (Jul. 2008), 1179–1189.

[7] Chang, Y. C., Chang, Y. W, Wu, G. M., and Wu, S. W. 2000. B-trees: A new representation for nonslicing floorplans. In *Proc. 37th ACM/IEEE Design Automation Conf.* (2000), 458–463.

[8] Cohn, J. M., Garrod, D. J., Rutenbar, R. A., and Carley, L. R. 1991. KOAN/ANAGRAM II: New Tools for Device-Level

Analog Placement and Routing. *IEEE Journal of Solid-State Circuits.* 26, 3 (Mar. 1991) 330 - 342.

[9] Gielen, G. G. E., and Rutenbar, R. A. 2000. Computer-aided design of analog and mixed-signal integrated circuits. *Proceedings of the IEEE.* 88, 12 (Dec. 2000), 1825-1852.

[10] Gielen, G. G. E. 2005. CAD tools for embedded analogue circuits in mixed-signal integrated systems on chip. *IEE Proceeding on Computers and Digital Techniques*. 152, 3 (May 2005), 317–332.

[11] Graeb, H. E. 2010. *Analog Layout Synthesis: A Survey of Topological Approaches*. Springer.

[12] Guo, P. N., Cheng, C.-K., and Yoshimura, T. 1999. An O-tree representation of nonslicing floorplan and its applications. In *Proc. 36th ACM/IEEE Design Automation Conf.* (1999), 268–273.

[13] Habal, H., and Graeb, H. 2011. Constraint-Based Layout-Driven Sizing of Analog Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.* 30, 8 (Aug. 2011), 1089-1102.

[14] Jai-Ming, L., and Yao-Wen, C. 2001. TCG: a transitive closure graph-based for non-slicing floorplans. In *Proc. 38th ACM/IEEE Design Automation Conf.* (2001), 764-769.

[15] Jangkrajarng, N., Bhattacharya, S., Hartono, R., and Shi, C. 2003. IPRAIL-Intellectual property reuse-based analog IC layout automation. *Integration, VLSI Journal*. 36, 4 (Nov. 2003), 237–262.

[16] Jingnan, X., Vital, J., and Horta, N. 2001. A SKILLTM - based Library for Retargetable Embedded Analog Cores. *Proceedings Design, Automation and Test in Europe* (Mar. 2001), 768-769.

[17] Khademsameni, P., and Syrzycki, M. 2002. A tool for automated analog CMOS layout module generation and placement. In *IEEE Canadian Conference on Electrical and Computer Engineering*. 1 (May 2002), 416 - 421.

[18] Koda, S., Kodama, C., and Fujiyoshi, K. 2007. Linear programming-based cell placement with symmetry constraints for analog IC layout. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 26, 4 (Apr. 2007), 659–668.

[19] Lampaert, K., Gielen, G., and Sansen, W. 1995. A performance-driven placement tool for analog integrated circuits. *IEEE Journal of Solid-State Circuits*. 30, 7 (Jul. 1995), 773–780.

[20] Lin, P. H., and Lin, S. C. 2007. Analog placement based on novel symmetry-island formulation. In *Proc. 44th ACM/IEEE Design Automation Conf.* (2007), 465–470.

[21] Lin, P. H., and Lin, S. C. 2008. Analog placement based on hierarchical module clustering. In *Proc. 45th ACM/IEEE Design Automation Conference* (Jun. 2008), 50–55.

[22] Lin, P. H., Zhang, H., Wong, M., and Chang, Y. W. 2009. Thermal-driven analog placement considering device matching. In *Proc. 46th ACM/IEEE Design Automation Conf.* (Jul. 2009), 593-598.

[23] Lourenço, N., Vianello, M., Guilherme, J., and Horta, N. 2006. LAYGEN - Automatic Layout Generation of Analog ICs from Hierarchical Template Descriptions. In *Conference on Ph.D. Research in Microelectronics and Electronics*. (Jun. 2006), 213-216.

[24] Lourenço, N., and Horta, N. 2012. GENOM-POF: Multi-Objective Evolutionary Synthesis of Analog ICs with Corners Validation. In *Genetic and Evolutionary Computation Conference* (Jul. 2012) Philadelphia, USA.

[25] Malavasi, E., Charbon, E., Felt, E., and Sangiovanni-Vincentelli, A. 1996. Automation of IC layout with analog constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 15, 8 (Aug. 1996), 923–942.

[26] Mentor Graphics, http://www.mentor.com/.

[27] Meyer, V. 1993. ALSYN: Flexible rule-based layout synthesis for analog ICs. *IEEE J. Solid-State Circuits*. 28, 3 (Mar. 1993), 261–268.

[28] Nakatake, S., Fujiyoshi, K., Murata, H., and Kajitani, Y. 1998. Module packing based on the BSG-structure and IC layout applications. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 17, 6 (Jun. 1998), 519–530.

[29] Ranjan, M., Verhaegen, W., Agarwal, A., Sampath, H., Vemuri, R., and Gielen, G. 2004. Fast, layout inclusive analog circuit synthesis using pre-compiled parasitic-aware symbolic performance models. In *Design Automation Conference and Test in Europe Conference*. 1 (Feb. 2004), 604–609.

[30] Rijmenants, J., Litsios, J., Schwarz, T., and Degrauwe, M. 1989. Ilac: An automated layout tool for analog cmos circuits. *IEEE Journal of Solid-State Circuits*. 24, 2 (Apr. 1989), 417–425.

[31] Rutenbar, R. A. (2010) Analog layout synthesis: What's missing?. In *Proceedings of the 19th international symposium on Physical design.* (Jan. 2010), 43.

[32] Strasser, M., Eick, M., Gräb, H., Schlichtmann, U., and Johannes, F. M. 2008. Deterministic analog circuit placement using hierarchically bounded enumeration and enhanced shape functions. In *IEEE/ACM International Conference on Computer-Aided Design* (Nov. 2008), 306-313.

[33] Suman, B., and Kumar, P. 2006. A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the Operational Research Society.* 57 (2006), 1143-60.

[34] Vancorenland, P., der Plas, G. V., Steyaert, M., Gielen, G., and Sansen, W. 2001. A layout-aware synthesis methodology for RF circuits. In *IEEE/ACM International Conference on Computer-Aided Design* (Nov. 2001), 358 – 362.

[35] Yilmaz, E., and Dundar, G. 2009. Analog Layout Generator for CMOS Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 28, 1 (Jan. 2009), 32–45.

[36] Zhang, L., Kleine, U., and Jiang, Y. 2006. An Automated Design Tool for Analog Layouts. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*. 14, 8 (Aug. 2006), 881–894.

[37] Zhang, L., and Liu, Z. 2010. A Performance-Constrained Template-Based Layout Retargeting Algorithm for Analog Integrated Circuits. In *Proc. 47th ACM/IEEE Design Automation Conference* (Jan. 2010), 293 - 298.