GENERATION OF TECHNOLOGY-PORTABLE FLEXIBLE ANALOG BLOCKS

R. Castro-López, F. V. Fernández, M. Delgado-Restituto, F. Medeiro and A. Rodríguez-Vázquez
Instituto de Microelectrónica de Sevilla, Centro Nacional de Microelectrónica
Edif. CICA, Avda. Reina Mercedes s/n, E-41012 Sevilla, SPAIN
Phone: +34 955 056 666 FAX: +34 955 056 686 e-mail: castro@imse.cnm.es

ABSTRACT

This paper introduces a complete methodology for retargeting of analog blocks to different sets of specifications, even to different technology processes. By careful integration of the tuning process of design parameters with layout generation, fully functional designs are generated in a few minutes of CPU time.

1. INTRODUCTION

For an increasing number of designers, the secret to quickly building highly integrated systems on a chip in a shrinking development cycle lies in the extensive reuse of intellectual property modules. While a lot of progress has been made in the digital arena in recent years, the specific characteristics of analog design makes the development of flexible analog modules a much more difficult task.

To contribute to the solution of this problem, this paper proposes a retargeting for reusability methodology for analog blocks, able to provide working designs for each new set of specifications and/or for different technology processes. The objective of this methodology is not to get the optimum design but to get a design which meets the required specifications in the shortest time. Parameterized layout templates have been used to address different design specifications, together with a new methodology to instantiate the template in any technology process, even from different foundries. This is extremely important given the fast pace of technology evolution. Design parameters are tuned to each new set of specs (in the same or another technology process) by combining electrical simulation with statistical and/or deterministic optimization. The tuning engine is also able to handle circuit-specific knowledge, previously introduced using a common programming language. Size tuning is also integrated with the layout by concurrently estimating the quality of the instantiated layout.

The proposed methodology is discussed in Section 2. Sections 3 and 4 are devoted to its two main components: the layout generator and the size tuning tool. Finally, Section 5 illustrates the retargeting of a fully-differential operational amplifier.

2. RETARGETING METHODOLOGY

The proposed retargeting methodology relies on the construction of: a circuit topology description, technology-portable parameterized layout templates, and, possibly, some tuning strategies in the form of design constraints. Retargeting of a given block for a new set of specifications is performed using the iterative mechanism illustrated in Fig.1.

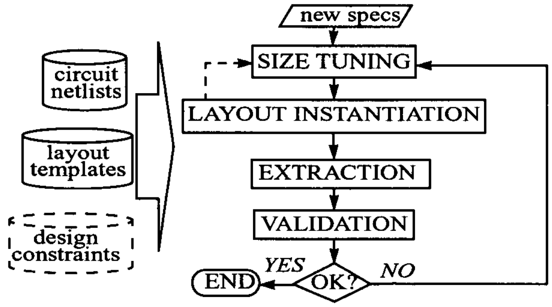


Figure 1. Retargeting cycle.

Given a new set of specifications, device sizes must be tuned to achieve such specifications. Size tuning is based on an appropriate combination of design rules and constraints, and optimization-based sizing using electrical simulation. Resulting device sizes are instantiated on the layout template. Layout parasitics are extracted and the circuit performances validated through electrical simulation. In case some specs are not met, validation information is fed back to the size tuning engine to perform a new iteration.

A single iteration of the retargeting methodology is usually sufficient because of: (a) the use of a simulation-based approach in the tuning procedure, which guarantees accuracy of the predicted performances; and, (b) the inclusion of layout constraints and parasitic estimations in the tuning procedure.

3. GENERATION OF TECHNOLOGY-PORTABLE LAYOUT TEMPLATES

Basically, there are two groups of approaches to automated layout of analog blocks: (a) those based on capturing expert knowledge in the form of templates or procedures, and (b) those based on its formulation as an optimization problem. Taking into account that our objective is to build circuit-specific layout generators and that reuse of designers' expertise is a major concern we have opted for a parameterized template approach. Although its main drawbacks are their smaller flexibility and the relatively high cost of the template/procedure generation for each block, this is a better approach for our objectives than that based on for-

0-7803-7448-7/02/$17.00 ©2002 IEEE

mulating the layout generation as an optimization problem, where computational cost is considerably higher and the addition of designers' expertise is quite difficult. To palliate the flexibility problem a complete hierarchical parameterization has been pursued together with a strong coupling with the tuning procedures. The template development cost has been reduced by a strong hierarchical decomposition that allows the reuse of subcell layout templates in larger templates; and by using appropriate structures.

The construction of technology-portable parameterized layout templates has been based on Parameterized Cells (pCells) and the SKILL programming language, from the CADENCE framework.

When parameterizing complex layout cells, factors such as regularity, density and symmetries must be kept during the retargeting process. This has been achieved by relying on a deep hierarchical decomposition and a careful cell planning. During this constructive process much attention is paid to the complete parameterization of cells, relative positions and interconnections, so that, wide changes in device sizes can be accommodated.

The main problems arising when migrating from one technology (T1) to another (T2) are:

- Only the layers equally defined in T2 and T1 and, therefore, playing the same role in both technologies are available in T2.
- Off-grid problems, since the grid depends on the technology.
- T1 may have layers T2 does not have. As a consequence of this, if we are using, for example, metal-3 in T1, and the layout is migrated to T2, where there is no metal-3 layer, then, changes on the original layout have to be made. This means that, if the original layout is not modified at all, the portability will only be possible between similar technologies.
- Process parameters (resistances, capacitances, etc.) vary.
- The design rules are violated.

The proposed solution for creating technology-portable parameterized cells is shown in Fig. 2 and proceeds as follows: (1) starting from the graphical pCell environment in the initial technology, the polygons are drawn and some basic commands (stretch lines, repetitions, inherited parameters and inherited layers) to create the fundamental parameters (width, length, multiplicity, etc.) of the cell, are compiled together with the pCell to generate its SKILL code; (2) then, using a generic design rule database, the numerical coordinates of the cell (whose polygons have been carefully planned to avoid the portability problems) are substituted by generic coordinates; (3) finally, the modified code is compiled back to a graphical pCell for each technology process. The code calls back to a technology rule database when a generic coordinate is read and takes from that database the value of the coordinate. In this way, the cells are laid-out in different technologies with no errors.

The construction of a fully-parameterized layout template able to accommodate very different device sizes and technology portable requires much more effort than the creation of a full-custom layout for a sized circuit, typically about a factor of five times more expensive. The instantiation of the layout template for each new set of device sizes takes less than one second of CPU time and requires no user interaction. Therefore, the additional development cost of the technology-portable layout template is largely compensated through its repetitive use.

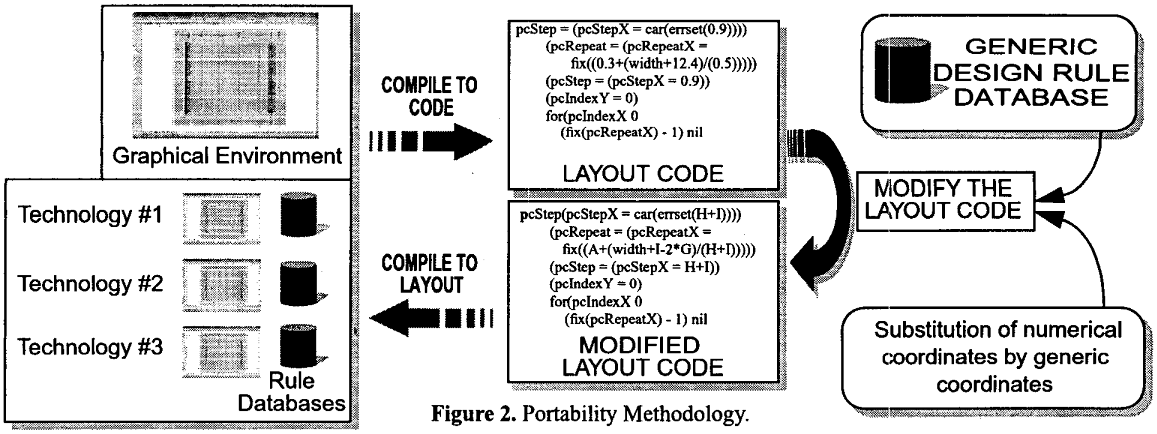
4. LAYOUT-CONSTRAINED SIZE TUNING

Design parameters must be appropriately tuned to meet the new design specifications: *restrictions* on the performance of a circuit, and/or *design objectives*. *Restrictions* are those specifications that include inequalities (i.e. DC-gain > 70 dB) and *design objectives* those whose intention is to maximize or to minimize some figure (i.e. minimize power consumption).

Basically two kinds of approaches have been formulated to the sizing problem: knowledge-based [1] and optimization-based [2]-[4]. Optimization-based approaches have become much more popular in recent years. The optimization process is an iterative procedure, design parameters being updated at each iteration until an equilibrium point is reached. Within optimization-based systems, simulation-based approaches are intrinsically open and the predicted performances have the accuracy of the electrical simulator used, at the price of a higher computational cost.

The degree of compliance of *restrictions* and *design objectives* at each iteration is quantified through a cost function. Mathematically, the fulfillment of these groups of specifications can be formulated as a constrained optimization problem with multiple *restrictions* and *objectives*:

$$\begin{array}{ll} \text{minimize} & y_{oi}(\mathbf{x}) \qquad , 1 \le i \le P \\ \text{subjected to} & y_{rk}(\mathbf{x}) \ge Y_{rk} \text{ or } y_{rk}(\mathbf{x}) \le Y_{rk}, 1 \le k \le R \end{array} \qquad (1)$$



```
pcStep = (pcStepX = car(errset(0.9))))
      (pcRepeat = (pcRepeatX =
           fix((0.3+(width+12.4)/(0.5)))))
      (pcStep = (pcStepX = 0.9))
      (pcIndexY = 0)
      for(pcIndexX 0
         (fix(pcRepeatX) - 1) nil
```

```
pcStep(pcStepX = car(errset(H+I))))
      (pcRepeat = (pcRepeatX =
         fix((A+(width+I-2*G)/(H+I)))))
      (pcStep = (pcStepX = H+I))
      (pcIndexY = 0)
      for(pcIndexX 0
         (fix(pcRepeatX) - 1) nil
```

Figure 2. Portability Methodology.

where **x** is the vector $\{x_1, x_2, ..., x_N\}$, defining a multidimensional design space, y_{oi} stands for the value of the i-th *design objective*; y_{rk} are the values of the *restriction*-type specifications; and Y_{rk} are the targeted values of such specifications. The cost function in our implementation is defined in minimax sense:

$$\Psi(\mathbf{x}) = \begin{cases} \left(-\sum_i w_i \log(|y_{oi}|)\right) & if\ \mathbf{x} \in R_A \\ max\left[-w_k \log\left(\frac{y_{rk}}{Y_{rk}}\right)\right] & if\ \mathbf{x} \notin R_A \end{cases} \quad (2)$$

where R_A (acceptability region) denotes the region of the parameter space where restrictions are met, w_i is the weight associated with the i-th design objective (a positive real number if y_{oi} is positive and vice versa) and w_k is the weight associated with the restrictions that must be a real positive (alternatively negative) if the restriction is of type $\leq$ (alternatively $\geq$). These weights are used to give priority to the fulfillment of their associated specifications.

The implemented simulation-based approach is a two-step one: in the first one, statistical optimization techniques (capable to escape from local minima thanks to a nonzero probability of accepting movements that increase the cost function at the price of a larger CPU time), inspired on simulated annealing, are applied, while deterministic ones (where the optimization process is quickly trapped in a local minimum of the cost function) are applied in the second [3].

Some innovative features in the generation and acceptance of movements through the design parameter space allow to drastically reduce the computational cost: first exploration of the design space using a coarse grid to determine the best regions for further exploration, adaptive control of the temperature in the statistical techniques, synchronization of movement amplitude in parameter space with the temperature, etc.

The addition of designers′ expertise on tuning procedures for a specific block has also been enabled. Such knowledge takes the form of arbitrary relationships between design parameters, and artificial *restrictions* and/or *design objectives*, and can be expressed using the full capabilities of the C programming language. During the optimization process such relationships are taken into account, and *restrictions* or *design objectives* handled as in Eq. 2. From this point of view, our size tuning procedure is an optimization-based system incorporating the appealing features of knowledge-based ones.

In general, the design space for a given circuit contains several acceptability regions and each point in these regions is associated to different values of the design objectives. The instantiation of such solutions of the design space in the layout template yields layout instances of different quality. As this is a major concern, a quality evaluation is performed at each iteration and contributes to control the evolution of the optimization process. This evaluation of layout quality is based on the study of the particular layout template. It includes from individual constraints based on pCell size variations with respect to a reference, to composite constraints involving a number of pCells. To illustrate this point, let us consider the partial template in Fig. 3, composed of 4 pCells. Let us denote f_i, a function which returns the width of the i-th pCell. That will depend on: the sizes of the devices composing that pCell (or, in general, sizes of the pCells at lower hierarchical levels), the possible variants for its implementation (i.e., the number of fingers in an interdigitized structure), technological parameters, etc. If we take f_1 as a reference, an example of individual constraint would be:

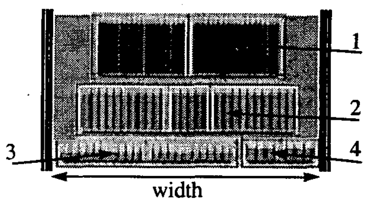


Figure 3. An example of layout constraints.

$$\text{minimize} \qquad abs(f_1 - f_2) \qquad (3)$$

A simple example of a composite constraint would be:

$$\text{minimize} \qquad abs[f_1 - (f_3 + f_4)] \qquad (4)$$

Once formulated in this way, layout constraints establish relationships between design parameters or constitute objective functions that the optimization engine can handle like the restrictions and design objectives in Eq. 2. Relative weights can also be used here to give priority to some layout constraints over others.

5. A RETARGETING EXAMPLE

The methodology presented in this paper will be illustrated via the retargeting of the fully-differential Miller-compensated two-stage amplifier in Fig. 4 for a different set of specifications. The circuit was originally designed for the set of specifications in Table 1. The design of the original circuit served to develop some circuit-specific design constraints (i.e., relationships of design parameters to ensure enough current in the current sources of the folded-cascode amplifier in the first stage to drive the transistors under maximum slew-rate conditions).

The sized circuit was also used as a reference to build a layout template. The cell layout was planned to allow a maximum

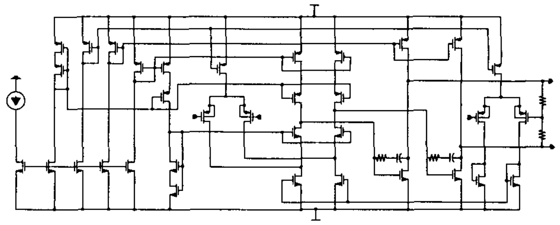


Figure 4. Fully-differential operational amplifier.

Table 1. Specs of original design.

	Specs	Simulated	Units
A_0	>80	87.8	dB
GBW	>35	35.1	MHz
PM	>45	47.2	o
OS	>5	5.3	V
power	minimize	1.73e-3	W

flexibility to the component devices to accommodate the needed new sizes in a cell retargeting process.

Then, the circuit was retargeted to the set of specifications in the first two columns in Table 2. To illustrate the importance of including the evaluation of the layout quality during the size tuning process, two retargeting experiences were carried out. In the first one, the only objective was to achieve the new set of specifications. The instantiation of the resulting device sizes in the layout template yielded the layout in Fig. 5. The extracted layout met the specifications but, as can be observed, the layout density has largely been deteriorated.

In the second retargeting experience the impact on the layout quality was included in the cost function guiding the optimization process. As can be observed in the instantiation of the sizes in the layout template in Fig. 6, the layout quality and the area efficiency are much better (both layouts in Fig. 5 and Fig. 6 have been captured with the same resolution, therefore, the figures show the actual relative dimensions of both cells). The simulation results of the extracted layout are shown in the third column in Table 2. Notice that the specifications are also met in this new retargeted design.

Only one iteration of the retargeting methodology was needed. The total CPU time of the retargeting process was 5 minutes on a SUN Ultra 10 at 333 MHz. Four of them were spent on the size tuning task while the rest was spent on layout instantiation, DRC, extraction, LVS and final simulation.

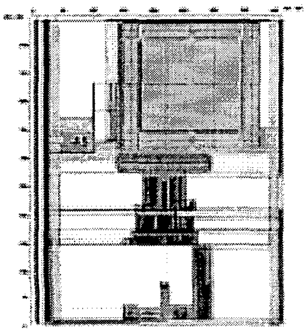


Figure 5. Instantiation of the retargeted design without evaluation of impact on layout quality.

Table 2. Specs and results of retargeted design in Fig. 6.

	Specs	Simulated	Units
A_0	>85	87.5	dB
GBW	>100	100.2	MHz
PM	>50	50.4	deg
OS	>5	5.01	V
power	minimize	1.93e-3	W

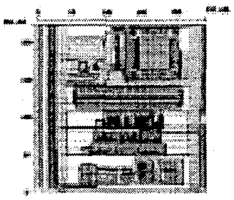


Figure 6. Instantiation of the retargeted design with evaluation of impact on layout quality.

As an example of retargeting to a different technology we will consider the migration of the original design to a 0.5μm technology of a different foundry. Table 3 shows the specifications in the new technology. The instantiation of the devices sizes in the technology-portable template yields the layout cell in Fig. 7. The simulation of the extracted layout provides the simulated results in the third column in Table 3.

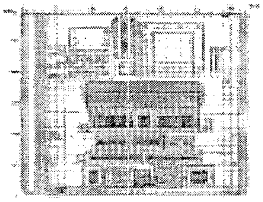


Figure 7. Instantiation of the retargeted design in a different technology process.

Table 3. Specs and results of retargeted design.

	Specs	Simulated	Units
A_0	>80	80.0	dB
GBW	>50	52.7	MHz
PM	>45	45.1	deg
OS	>5	5.2	V
power	minimize	9.1e-4	W

6. CONCLUSIONS

Lack of flexibility of analog blocks limit their applicability in application scenarios demanding high circuit performances. Through the introduction of a retargeting methodology for transistor level circuits, this paper has tried to contribute in making flexible technology-portable analog blocks a reality.

7. ACKNOWLEDGMENTS

This work has been partially supported by project TIC2001-0929 (funded by the Spanish Ministry of Science and Technology and the European Regional Development Fund).

8. REFERENCES

[1] M. Degrauwe et al., "IDAC: An Interactive Design Tool for Analog CMOS Circuits," *IEEE J. of Solid-State Circuits*, Vol. 22, No. 6, pp. 1106-1116, Dec. 1987.

[2] G. Gielen, H. Walsharts and W. Sansen, "Analog Circuit Design Optimization based on Symbolic Simulation and Simulated Annealing" *IEEE J. Solid-State Circuits*, Vol. 25, No. 3, June 1990.

[3] F. Medeiro, R. Rodríguez-Macías, F.V. Fernández, R. Domínguez-Castro, J.L. Huertas and A. Rodríguez-Vázquez, "Global Design of Analog Cells Using Statistical Optimization Techniques," *Analog Integrated Circuits and Signal Processing*, Vol. 6, No. 3, pp. 179-195, Nov. 1994.

[4] R. Phelps, M. Krasnicki, R.A. Rutenbar, L.R. Carley and J.R. Hellums, "Anaconda: Simulation-Based Synthesis of Analog Circuits Via Stochastic Pattern Search," *IEEE Trans. on Computer-Aided Design*, Vol. 19, No. 6, pp. 703-717, June 2000.