Chameleon ART: A Non-Optimization Based Analog Design Migration Framework

Sherif Hammouda[1,4], Hazem Said[2], Mohamed Dessouky[1], Mohamed Tawfik[1], Quang Nguyen[3], Wael Badawy[4], Hazem Abbas[1], Hussein Shahein[2]

[1] Mentor Graphics Egypt, 51 Beirut St. Heliopolis, Cairo, Egypt.

[2] Ain Shams University, 1 El-Sarayat St., Abbassia, Cairo, Egypt.

[3] ON Semiconductor France SAS, Toulouse, France

[4] University of Calgary, 2500 University Drive NW, Calgary, Alberta.

ABSTRACT

Presented in this paper is a tool that automatically migrates analog designs from one process to another while keeping circuit and layout topologies. A netlist migration engine recalculates the new device dimensions in the target technology followed by a layout migration engine that compacts the design according to the new process design rules. The overall framework preserves design intelligence embedded in the original IP such as symmetry, hierarchy, placement and routing. The circuit migration engine, being very fast, can retarget large analog blocks in only a few minutes while giving same or better performance of the original design. The migration of 3 different circuits is presented to validate the overall methodology. These circuits have been fabricated and measured.

Categories and Subject Descriptors:

B.7.2 [Integrated Circuits]: Design Aids

General Terms

Algorithms, Performance, Design.

Keywords

Analog Reuse, Design Extraction, Circuit Sizing, Layout Compaction, Layout Retargeting.

1. INTRODUCTION

Driven by the need to be more powerful, semiconductor manufacturers continue to innovate technologies towards smaller transistor feature sizes. As a result, there is an increasing need in re-designing functioning mixed-signal designs for new processes. Analog IP reuse is still a manual process where the analog IP is redesigned each time an SoC is migrated to a new technology.

This paper presents a complete framework, that retargets hard analog IPs through different processes in a fully automated way without the use of time consuming optimization techniques.

DAC 2006, July 24–28, 2006, San Francisco, California, USA.

The rest of the paper is organized as follows: section 2 presents an overview of the state of the art for analog circuit migration techniques, section 3 and 4 present the suggested circuit-sizing and layout retargeting approaches, section 5 presents examples and results and finally section 6 presents a summary and draws conclusions of the work presented.

2. ANALOG DESIGN MIGRATION

Several approaches have been investigated to resize analog cores. One approach consists of developing specific block synthesis tools [1] which take a considerable effort and time. Therefore it must be justified by an extensive use of the generator.

Design reuse based on an original *working* design has been investigated both through qualitative reasoning [2] and analog synthesis [3][5]. Another approach optimizes the equivalent small-signal model with respect to the original circuit [4]. The optimization engine visits candidate circuit designs and adjusts their parameters in an attempt to satisfy their user's specified performance goals [3]. A first group of optimization techniques use analytical models that describe the basic performance of the circuit using symbolic equations [5]. A second group of optimization techniques uses the full spice accuracy [3]. Optimization techniques in general are closer to a *circuit re-design* than *design reuse*. All design knowledge and tradeoffs, implicitly coded by the first designer in the initial design, is completely lost. In addition, optimization-based techniques are only adapted to cell sizing due to their extensive use of computer resources. They are therefore less suitable for migrating a complete mixed-signal function (e.g. analog-to-digital converter, PLL, …) .

3. CIRCUIT SIZING

During circuit retargeting, the main focus is to keep the same circuit performance of a given circuit while trying to migrate it. This is done hierarchically for the whole macro-function. Naturally, if the performance of each block is kept the same during design migration, the performance of the whole macro-function will also be the same.

The structure of the netlist migration engine is shown in Figure 1. The input to this engine consists of information about the source

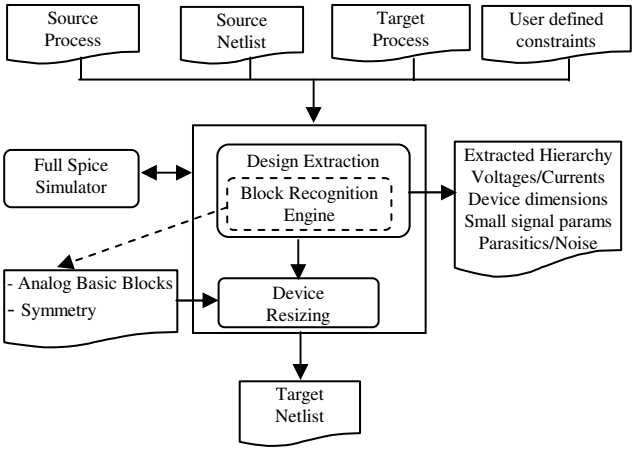

Figure 1. Structure of the netlist migration engine

and target processes, a hierarchical netlist file in spice format and user defined constraints such as fixing of some node voltages, minimum and maximum device dimensions. The main blocks for this engine are a design extraction engine, a block recognition engine and a device resizing engine.

As shown in Figure 1 the retargeting process itself is divided into two main actions: Design extraction and Device resizing, which are detailed in the following sections:

3.1 Design Extraction Engine

The design extraction engine extracts all necessary data using an analog simulator with the **full spice models**. The hierarchy of the design is extracted first, and then all device dimensions, currents, biasing voltages, small-signal parameters, parasitics and noise associated with the devices are extracted from each device in each subcircuit in the design. A block recognition engine is embedded within the Design Extraction Engine and is capable of extracting different analog basic building blocks such as: Current Mirror, Level Shifter, Voltage Reference, Current mirror Load, Differential Pair, Flip Flop, Current mirror block, Level Shifter Block, Current Source and Switch [6].

3.2 Device Resizing Engine

The resizing algorithm is based on the assumption that preserving the parameters of each individual component in each subcircuit would preserve the overall performance of each subcircuit.

The core of the algorithm is based upon the definition of a *relative* transistor bias point as follows:

$$I_{target} = KI_{source} \quad \text{(K is a scaling constant)} \tag{1}$$

$$V_{GS-relative} = V_{GS} - V_T \tag{2}$$

$$V_{DS-relative} = V_{DS} - V_{DSsat} \tag{3}$$

It is easy to show that, as a starting point, keeping the same relative bias during retargeting results in transistor parameters close to their source values. The methodology used to fine tune transistors parameters, then, depends on their functionalities. A sample of three different types of functionality is given in Table 1 to illustrate the device resizing algorithm.

Table 1 Parameters to resize according to block type

Block	Parameters to preserve	Comments
Differential Pair	$g_m \propto \frac{I_D}{V_{GS}-V_T}$ $g_{ds} \propto \frac{I_D}{L}$	Gate and source voltages are scaled to keep same relative bias point. Drain voltage is scaled to accommodate the supply to ground room
Cascoded Current Mirror	$R_o = g_{m4} r_{o4} r_{o2}$ R_o is the output resistance of he cascode	r_{o2} is the output resistance of the mirror transistor, while g_{m4} and r_{o4} are the transconductance and output resistance of the cascode transistor.
Switch	$g_{ds} \propto \frac{W}{L}(V_{GS}-V_T)$	g_{ds} is the conductance of the switch.

4. LAYOUT MIGRATION

Source layout contains valuable physical design knowledge, which is usually already verified through fabrication. Layout migration by compaction keeps the same knowledge (i.e. floor-plan, placement and routing), precious to analog designers [7].

4.1 Layout Migration Engine

The proposed layout migration engine is illustrated in Figure 2. Starting from a source layout a layer-mapping module converts the layers between processes. A device extraction module identifies physical locations and device parameters while a constraint generation module extracts design rule constraints and layout connectivity. The new geometrical device parameters generated by the netlist migration engine are also converted to layout constraints.

4.2 Layout Compaction Module

Most existing compaction techniques use one-dimensional compaction, as in virtual grid [8], shear lines [9] and constraint graph [10] approaches. Constraint graph approaches are more flexible and generate better quality layouts [11].

The proposed layout compactor is based on a constraint graph model. Rather than using a symbolic approach, it handles edges of layout polygons directly. Figure 3 shows a sample layout together with the corresponding constraint graph. The constraint graph is a directed graph in which *nodes* represent edges of layout polygons, while *arcs* represent a distance constraint between two edges represented by the two end nodes of the arc. A distance can either

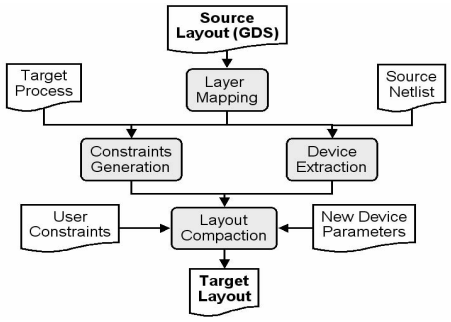

Figure 2. Layout Migration Engine

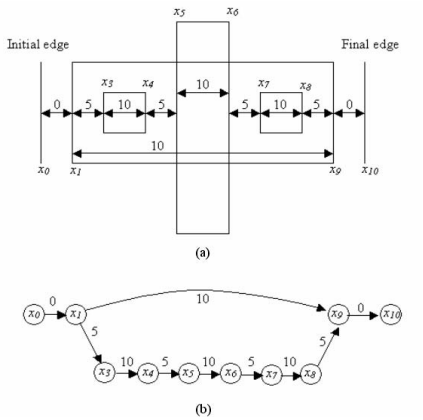

Figure 3. (a) sample layout and (b) the corresponding constraint graph

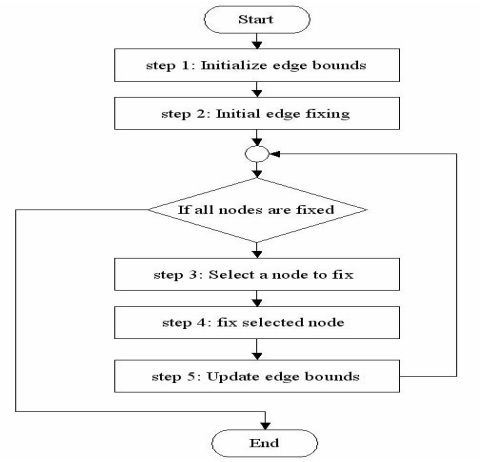

Figure 4. The steps of layout compaction module

be a width constraint or a separation constraint. Solving a constraint graph using the well-known longest path algorithm [9] results in undesired polygon extensions such as the wire-length minimization problem. The detailed steps of the proposed algorithm are shown in Figure 5:

Step 1: Initialize edge bounds

Minimum and maximum edge bounds are initialized as follows: The longest path algorithm [9] is used to compact the layout to the left (downward), to find *minimum* edge bounds then to the right (upward), to obtain the *maximum* edge bounds.

Step 2: Initial edge fixing

Edges having their bounds equal are marked as fixed edges. Paths having only fixed nodes are marked as fixed.

Step 3: Select a node to fix

For non-fixed nodes, a node is selected based on the ability to move, the difference between minimum and maximum bounds, and the corresponding polygon area reduction.

Step 4: Fix selected node

The node is moved in the direction that minimizes polygon area.

Step 5: Update edge bounds

After fixing a node, the bounds of all non-fixed affected edges are updated. The update is done by using the graph technique similar to the longest path algorithm but starting from the fixed node.

These steps are repeated till all graph nodes are fixed. The algorithm produces a compacted layout with no need for wire length minimization or post processing. Dealing with edges other than symbols enables the compaction engine to handle any complex device as well as any complex routing.

5. EXAMPLES.

Several industrial test cases have been identified to validate the migration tool. The overall migration CPU time of the largest design took less than one hour on an Ultra Sparc machine.

5.1 Folded Cascode Amplifier.

The migration of a 2 stage folded cascode amplifier with cascode compensation (complex poles) is presented as a mean to compare our engine with NeoCircuit [3]. The design was migrated from a 0.35µm to a 0.18µm technology. The sizing of the circuit took around 10 minutes of CPU on an Ultra 10 Sparc machine compared to a 50 minutes run on a cluster of 19 workstations using optimization [3]. Table 2 presents the simulation results of the source and target technology.

Table 2 Source and Migrated specifications of the Amplifier

Technology	0.35u/3.3V (source)	0.18u/3.3V (migrated)
Gain	95.8 dB	101.5 dB
Phase Margin	60°	65.5°
Gain Margin	3.67	4.73
Unity Gain Freq.	13.25 MHz	21 MHz
PSRR	94 dB	122 dB
CMRR	92.1 dB	93.9 dB
Slew Rate	4 V/µ	4.6 V/µ
Settling Time	101ns	73ns
Load Capacitance	10 pF	10 pF
Power Consumption	3.3 mW	3.3 mW

5.2 100 nA High Precision Current Source.

The current source consists of around 140 transistors. Its current has a low variation with both temperature and supply voltage. It can operate from a supply voltage range: 2.5V to 5.5 V and in a temperature range: -55 to 125 °C. This design has been migrated successfully, fabricated and measured. Figure 5 shows simulation results of the output current variation over a variation in the output voltage and a temperature sweep. Figure 6 shows source and target layout with area saving of around 50% in silicon. The retargeting of this design took around 15 minutes of CPU time on an Ultra 10 Sparc machine.

5.3 Integrated Precision Crystalless Oscillator

This is an integrated astable oscillator that produces a 1 MHz output frequency. It has an output frequency with high stability versus temperature sweep from –55 to 125°C and with supply variations from 2.5 to 5.5V. The oscillator contains several blocks such as a bandgap reference, some biasing cells, a digital decoder used for trimming, an amplifier and a couple of comparators. The number of devices inside this circuit is around 500. The migration of the design from 0.6u to 0.25u took 1 week including corner simulation and post layout verification. Figure 7 shows simulation results of the oscillator versus temperature and supply voltage sweep respectively. Figure 8 shows the source and the retargeted layout with an area saving of around 8% which is less than the previous examples due to the large area occupied by passive devices.

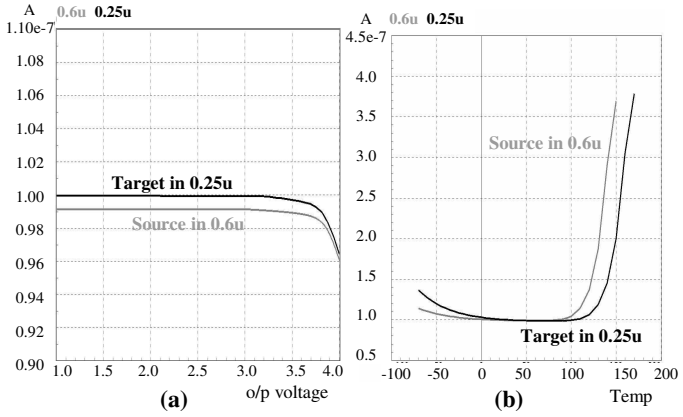

Figure 5. Bias current vs (a) output voltage (b) Temperature

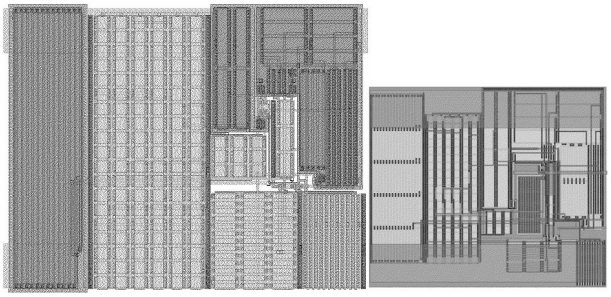

Figure 6. Bias Cell retargeting from 0.6u to 0.25u

6. CONCLUSION

This paper describes an innovative method for the reuse of analog circuits which includes both a circuit sizing engine and a layout-retargeting engine. The circuit-sizing engine is based on design extraction and device performance mapping. The layout engine is based on a modified edge-based compaction algorithm. Both netlist and layout engines are efficient with small cells as well as macro-functions and have proven to be very fast. The validation of the tool has been demonstrated by retargeting and fabricating several real designs through an industrial partnership.

7. REFERENCES

[1] R.R. Neff, P.R Gray, and A. Sangiovanni-Vincentelli, "A Module Generator for High-Speed CMOS Current Output Digital/Analog Converters", in *IEEE J. of Solid State Circuits*, Vol. 31, pp. 448-451, Mar. 1996.

[2] K. Francken, and G. Gielen, "Methodology for Analog Technology Porting Including Performance Tuning", in *Proc. IEEE Int. Symp. On Circuits and Systems*, Vol. 1, pp. 415-418, May 1999.

[3] E. Hennig, R. Sommer and L. Charlack, "An Automated Approach for Sizing Complex Analog Circuits in a Simulation-Based Flow", in *Proc. ACM/IEEE Design Automation and Test in Europe*, Designer Forum, pp. 230-234, Mar. 2002.

[4] S. Funaba, A. Kitagawa, T. Tsukada, and G. Yokomizo, "A Fast and Accurate Method of Redesigning Analog Subcircuits for Technology Scaling", Analog Integrated Circuits and Signal Processing, Kluwer Academic Publishers, Vol. 25, 2000, P. 299-307

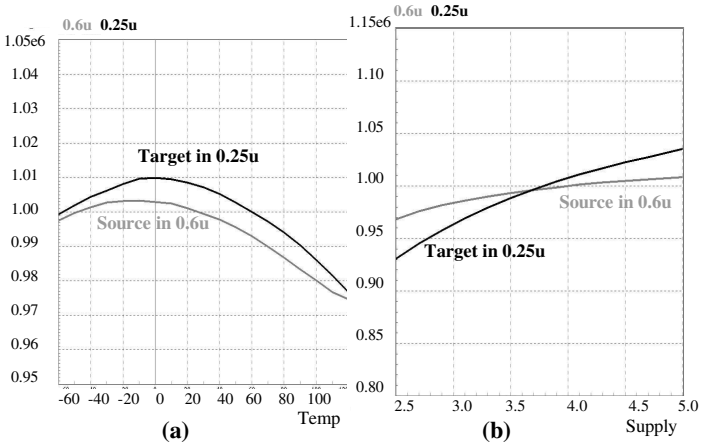

Figure 7. Oscillator Frequency vs (a) Temperature (b) Supply

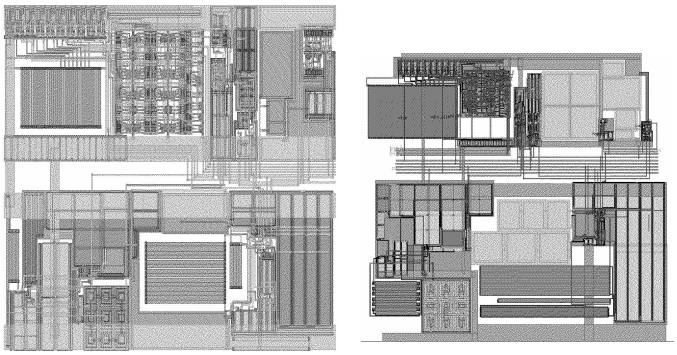

Figure 8. Oscillator retargeting from 0.6u to 0.25u

[5] Y. Xu, K-L. Hsiung, X. Li, I. Nausieda, S. Boyd, and L. Pileggi, "OPERA: Optimization with Ellipsoidal uncertainty for Robust Analog IC design", in *Proc. 42nd Design Automation Conf.*, June 2005, pp. 632–637.

[6] H. Graeb, S. Zizala, J. Eckmueller, and K. Antreich, "The Sizing Rules Method for Analog Integrated Circuit Design", in *Proc. Int. Conf. on Computer-Aided Design*, pp. 343 – 349, 2001.

[7] S. Bhattacharya, N. Jangkrajarng, R. Hartono, and C-J. R. Shi, "Correct-by-Construction Layout-Centric Retargeting of Large Analog Designs", in *Proc. Design Automation Conference*, June 2004, pp. 139-144.

[8] Bayer, D-G.; and Weste, N. "Virtual Grid Compaction Using the Most Recent Layers Algorithm", in *Proc. Int. Conf. on Computer-Aided Design*, 1983, pp. 92-93.

[9] A.E. Dunlop, "SLIM - The Translation of Symbolic Layouts into Mask Data", in *Proc. Design Automation Conference*, 1980, pp. 595 – 602

[10] Y. E. Cho, "A subjective review of compaction", in *Proc. Design Automation Conf.*, June 1985, pp. 396–404.

[11] Jianwen Zhu, Fang Fang, and Qianying Tang, "Calligrapher: A new layout-migration engine for hard intellectual property libraries," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, pp. 1347–1361, September 2005.