Analog Circuits Sizing Using the Fixed Point Iteration Algorithm with Transistor Compact Models

Farakh Javid, Ramy Iskander, François Durbin and Marie-Minerve Louërat

***Abstract*—This paper presents an algorithm, based on the fixed point iteration, to solve for sizes and biases using transistor compact models such as BSIM3v3, BSIM4, PSP and EKV. The proposed algorithm simplifies the implementation of sizing and biasing operators. Sizing and biasing operators were originally proposed in the *hierarchical sizing and biasing methodology* [1]. They allow to compute transistors sizes and biases based on transistor compact models, while respecting the designer's hypotheses. Computed sizes and biases are accurate, and guarantee the correct electrical behavior as expected by the designer. Sizing and biasing operators interface with a Spice-like simulator, allowing possible use of all available compact models for circuit sizing and biasing over different technologies. A *bipartite graph*, that contains sizing and biasing operators, is associated to the design view of a circuit, it is the design procedure for the given circuit. To illustrate the effectiveness of the proposed fixed point algorithm, a folded cascode OTA is efficiently sized with a 130nm process, then migrated to a 65nm technology. Both sizing and migration are performed in a few milliseconds.**

***Index Terms*—Analog IP, analog sizing, design reuse, bipartite graphs, transistor compact models, technology migration.**

I. Introduction

OVER the past few decades, research in analog synthesis led to the emergence of two major schools : the first school pushing towards *Full Design Automation (FDA)* and the second school pushing towards *Full Design Handcrafting (FDH)*. Many academic and commercial tools have been introduced by the FDA school such as OASYS [2], IDAC [3], OPASYN [4], DELIGHT.SPICE [5], ASTRX/OBLX [6], AMGIE [7], MAELSTROM [8], ANACONDA [9]. Except for OASYS, IDAC and AMGIE which are knowledge-based, the tools were mainly simulation-based. On the other hand, few FDH school academic tools have been introduced, such as COMDIAC [10], PAD [11], [12] and [13], which provide analog designers with sufficient insight for full trade-offs optimization.

Nowadays, many EDA companies develop tools that help to explore design trade-offs. To assess this research direction, we developed the *hierarchical sizing and biasing methodology* [1] to hierarchically size and bias analog circuits. This methodology generates suitable design procedures that respect the circuit topology and designer's constraints. It elaborates an intermediate design representation, called *bipartite dependency graphs*, that are used to represent the design procedures. These formal procedures are consistent, reusable, and technology independent.

Farakh Javid, Ramy Iskander and Marie-Minerve Louërat are with the LIP6 Laboratory, University Pierre and Marie Curie, Paris, France. e-mail: farakh.javid@lip6.fr

François Durbin is with the CEA DAM/DIF, Bruyères-le-Châtel, France.

The hierarchical sizing and biasing methodology consists of three main tasks: *transistor sizing and biasing*, *device sizing and biasing* and *circuit sizing and biasing*. Transistor sizing and biasing is performed using computational routines called *sizing and biasing operators*. These operators are used to size and bias elementary transistors. Device sizing and biasing is performed by constructing dedicated device sizing procedures based on elementary transistor operators. Circuit sizing and biasing combines sizing procedures of children devices or lower level subcircuits in order to construct the sizing procedure of the whole circuit. The circuit sizing procedure is an enumerated sequence of sizing and biasing operators of all devices forming the circuit.

In this paper, we focus on the transistor sizing and biasing task. This task was originally performed using Newton-Raphson algorithm as suggested in [1], [14]. We propose a new formulation of the sizing and biasing operators using derivative-free fixed point iteration. We construct and evaluate a complete sizing procedure in the form of a *bipartite graph* for a folded cascode amplifier. We show that industrial transistor compact models such as BSIM3v3, BSIM4, PSP and EKV can be directly used with the fixed point algorithm for the bipartite graph evaluation.

The paper is organized as follows : section II presents the motivation of this work. Section III recalls the context of this work, namely the *hierarchical sizing and biasing methodology*. Section IV details the fixed point iteration algorithm implementation within sizing and biasing operators. In section V, sizing and biasing operators implementing the fixed point iteration are applied on the design of a folded cascode amplifier. The paper is concluded in section VI.

II. Motivation : Analog Sizing Challenge

Analog sizing is composed of two tasks : the sizing itself and the verification, as shown in Fig. 1. The sizing task is usually performed manually, *i.e.* the designer extracts the design equations for a circuit and solves them using simplified device models. On the contrary, the verification task is done with a simulator and standard device models like BSIM3v3, BSIM4, PSP, EKV. Thus, the gap between the accuracy of the device models used during the sizing and verification steps implies numerous and complex iterations between these steps, in order to guarantee the circuit suitable electrical behavior with respect to the target operating point. Moreover, it is to be noted that this gap increases with newer technologies that take into account advanced physical effects [15]. Thus, the analog sizing challenge can be stated as how to perform accurate

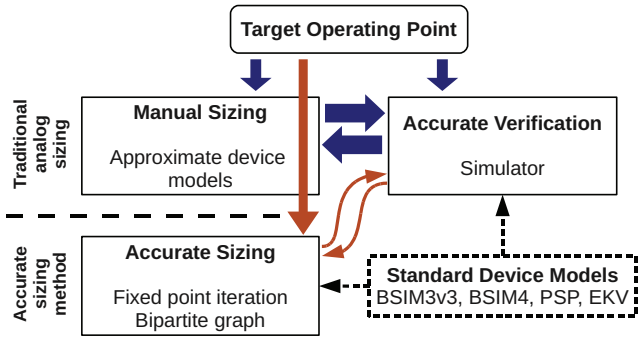

Fig. 1. Traditional analog sizing *vs* the proposed accurate sizing method.

sizing using standard device models. This would significantly reduce the iterations between the sizing and verification steps, and allow the designer to interactively explore design trade-offs.

In this work, we propose to use an efficient algorithm (the fixed point iteration) implemented within the sizing and biasing operators that interface with a simulator. Thus it is offered to the designer to size, in a relative short time, an analog circuit with the accuracy of any existing standard device model. This should reduce the iterations in the traditional analog sizing scheme, thus the sizing and verification are to be highly accelerated in our approach.

III. Hierarchical Sizing and Biasing Methodology

In this section, the hierarchical sizing and biasing methodology is recalled. It is based on a hierarchical representation of the circuit (section III-A), a library of *sizing and biasing operators* (section III-B) and a *bipartite graph* (section III-C).

A. Circuit Hierarchy

An electrical circuit is built as a hierarchy of interconnected subcircuits. A leaf subcircuit in the hierarchy is called a *device* : it is a set of transistors that realize an electrical function (like a differential pair or a current mirror). A *reference transistor* is defined in each device, it is the only transistor to be sized by an operator. In a circuit, the set of reference transistors corresponds to the well-known half-circuit. Computed sizes and biases for a reference transistor are then propagated, through designer constraints, to the other transistors in the device.

B. Sizing and Biasing Operators

1) Principle: Sizing and biasing operators are based on the inversion of the transistor compact model. Each operator has a set of input parameters that are set by the designer, and computes unknown widths and biases (see Table I, where $V_{EG} = V_{GS} - V_{TH}$). An operator computes either :

$$W = f_W(Temp, I_D, L, V_{GS}, V_{DS}, V_{BS}) \quad (1)$$

either :

$$V_{GS} = f_{V_{GS}}(Temp, I_D, W, L, V_{DS}, V_{BS}) \quad (2)$$

f_W and $f_{V_{GS}}$ are two inverse functions of the transistor compact model given below :

$$I_D = f_{MODEL}(Temp, W, L, V_{GS}, V_{DS}, V_{BS}) \quad (3)$$

where *MODEL* is a standard transistor model like BSIM3v3 [16], BSIM4 [16], PSP [17] and EKV [18]. f_W and $f_{V_{GS}}$ are monotonic functions, thus equations (1) and (2) are currently solved with the Newton-Raphson method. Convergence criteria for the Newton-Raphson method are the same as in commercial simulators.

TABLE I
Class definition of sizing & biasing operators

Operator	Definition
$OPVS(V_{EG}, V_B)$	$(Temp, I_D, L, V_{EG}, V_D, V_G, V_B) \mapsto (V_S, W, V_{TH})$
$OPVS(V_{GS}, V_B)$	$(Temp, I_D, L, V_{GS}, V_D, V_G, V_B) \mapsto (V_S, W, V_{TH})$
...	...
$OPVG(V_{EG})$	$(Temp, I_D, L, V_{EG}, V_D, V_S) \mapsto (V_G, W, V_{TH}, V_B)$
$OPVG(V_{GS})$	$(Temp, I_D, L, V_{GS}, V_D, V_S) \mapsto (V_G, W, V_{TH}, V_B)$
...	...
$OPVGD(V_{EG})$	$(Temp, I_D, L, V_{EG}, V_S) \mapsto (V_G, V_D, W, V_{TH}, V_B)$
$OPVGD(V_{GS})$	$(Temp, I_D, L, V_{GS}, V_S) \mapsto (V_G, V_D, W, V_{TH}, V_B)$
...	...
$OPW(V_{EG})$	$(Temp, I_D, L, V_D, V_{EG}, V_S) \mapsto (W, V_{TH}, V_B)$
$OPW(V_G, V_S)$	$(Temp, I_D, L, V_D, V_G, V_S) \mapsto (W, V_{TH}, V_B)$
...	...
$OPID(V_{EG})$	$(Temp, W, L, V_D, V_{EG}, V_S) \mapsto (I_D, V_{TH}, V_B)$
$OPID(V_G, V_S)$	$(Temp, W, L, V_D, V_G, V_S) \mapsto (I_D, V_{TH}, V_B)$
...	...

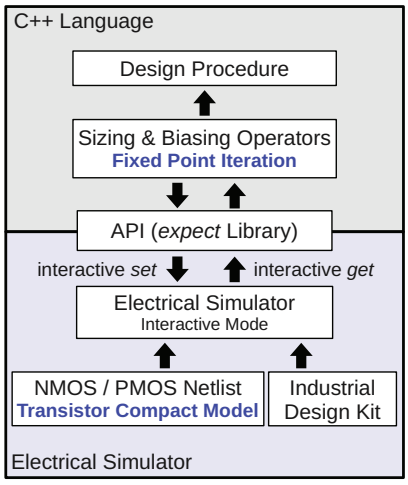

Fig. 2. Simulator encapsulation within sizing and biasing operators.

2) Simulator Encapsulation: As shown in Fig.2, an electrical simulator is encapsulated within the sizing and biasing operators [14]. Thus the sizing operators directly interface with industrial design kits to ensure the accuracy of the computed results. At the bottom in Fig.2, there is an electrical netlist that specifies the suitable design kit and contains only two transistors : one PMOS and one NMOS. Both transistors refer to a transistor compact model and are entirely sizable and biasable through simulator interactive commands. This two-transistor netlist is loaded by the electrical simulator launched in interactive mode, to perform sizing and biasing. Three types

of interactive commands are evaluated : *set*, *get* and *run*. The *set* command allows to set transistor known parameters (sizes and biases) at simulator level. The *get* command enables to retrieve currents, voltages and small signal parameters computed by the simulator. After a *set* command, a simulation must be run using the *run* command, in order to compute the DC operating point of the transistor. An API (Application Programming Interface) is developed using *expect* library [19] to automate the *set*, *get* and *run* commands execution. The API is used within sizing and biasing operators, that implement the fixed point iteration described in section IV. Sizing and biasing operators are optimized to minimize the number of calls to the simulator.

3) Operator Implementation: Operator $OPVG(V_{EG})$ implementation is illustrated in Fig. 3. First, transistor voltages and width are initialized in lines 5 to 13. *simulator* is the C++ object that encapsulates an electrical simulator. In line 8, the threshold voltage V_{TH} is retrieved from the simulator using *getVth()* function, that invokes the corresponding interactive command for the simulator. Lines 14 to 31 are for the width computation loop, using the fixed point iteration algorithm. On line 22, *runSimulation()* function performs a single transistor simulation to compute the operating point corresponding to the width computed in line 20. Computed width and voltages are returned in line 32.

1 **Operator** OPVG(V_{EG})
2 **Inputs** $Temp$, I_D, L, V_{EG}, V_D, V_S
3 **Outputs** V_G, V_B, V_{TH}, W
4 **Implements**
5 $V_B = V_S$
6 $V_{DS} = V_D - V_S$
7 $V_{BS} = V_B - V_S = 0.0$
8 V_{TH} = simulator.getVth()
9 $V_G = 0.0$
10 $W = 10 \cdot W_{min}$
11 simulator.setVds(V_{DS})
12 simulator.setVbs(V_{BS})
13 iteration_count = 0
14 **Do**
15 $W_{prev} = W$
16 $V_{G,prev} = V_G$
17 $V_G = V_S + V_{EG} + V_{TH}$
18 $V_{GS} = V_G - V_S$
19 simulator.setVgs(V_{GS})
20 Solve for W using the fixed point iteration
21 simulator.setW(W)
22 simulator.runSimulation()
23 V_{TH} = simulator.getVth()
24 Increment iteration_count
25 **While** (
26 ($\mid V_G - V_{G,prev} \mid \geq \epsilon_{reltol} \cdot max(\mid V_G \mid, \mid V_{G,prev} \mid) + \epsilon_{abstol,v}$
27 **or**
28 $\mid W - W_{prev} \mid \geq \epsilon_{reltol} \cdot max(\mid W \mid, \mid W_{prev} \mid) + \epsilon_{abstol,w}$)
29 **and**
30 iteration_count $\leq$ MAX_ITERATIONS
31)
32 **Return** $[V_G, V_B, V_{TH}, W]$

Fig. 3. Implementation of operator OPVG(V_{EG}) with simulator calls.

C. Bipartite Graph

The hierarchical sizing and biasing methodology handles design parameters dependencies in the form of a *bipartite graph* [20]. The bipartite graph is the design procedure associated to the simplified design view (half-circuit) of an analog circuit, it is formally defined below.

Definition 1. A *graph* $G(V, E)$ consists of a *vertex set* V (also called *node set*), an *edge set* E, and a *relation* that associates with each edge two vertices.

Definition 2. A *bipartite graph* $G(V_1, V_2, E)$ is a graph whose vertices are divided into two disjoint sets V_1 and V_2 ($V_1 \cap V_2 = \emptyset$) such that each edge $e \in E$ connects a vertex $x \in V_1$ to a vertex $y \in V_2$.

Definition 3. A *directed acyclic graph* (DAG) is a directed graph with no directed *cycles* (closed chain of vertices). In a DAG, all edges are oriented and are called *arcs*.

Definition 4. A bipartite DAG is a bipartite directed graph with no directed cycles.

Definition 5. We define a bipartite DAG $G = (V_p, V_c, A)$ with the two disjoint sets V_p and V_c. V_p is the set of *parameter nodes*, V_c is the set of *computation nodes*, A is the set of arcs. Thus the bipartite DAG G represents the design procedure of an analog circuit. A parameter node in V_p is a geometrical or electrical parameter related to a reference transistor, a device or a circuit. V_p set is split into three subsets as below :

$$V_p = P_{in} \cup P_{out} \cup P_{prop} \tag{4}$$

with $P_{in} \cap P_{out} = \emptyset$, $P_{out} \cap P_{prop} = \emptyset$ and $P_{prop} \cap P_{in} = \emptyset$. P_{in} is the set of input parameters for the design procedure. P_{out} is the set of output parameters computed by the design procedure. P_{prop} gathers intermediate parameters propagated between successive computation nodes. A computation node is either a *sizing and biasing operator* (see section III-C1), a *linear constraint* (see section III-C2) or a *designer-defined procedure (DDP)* (see section III-C3). The parameters in P_{in}, the linear constraints and the DDPs constitute the designer tacit knowledge, they are set up according to the designer understanding of the circuit behavior, and are modified to reach the suitable circuit performances.

1) Reference Transistor Sizing and Biasing: To size and bias a reference transistor, a bipartite DAG is associated with it, using a unique sizing and biasing operator. The bipartite graph for the sizing and biasing of the diode-connected transistor using operator $OPVGD(V_{EG})$ (c.f. Table I) is shown in Fig. 4.

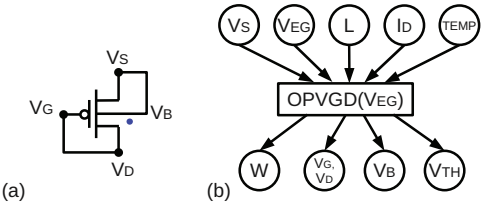

Fig. 4. (a) PMOS reference transistor (marked with a dot), (b) its associated bipartite graph with a sizing and biasing operator.

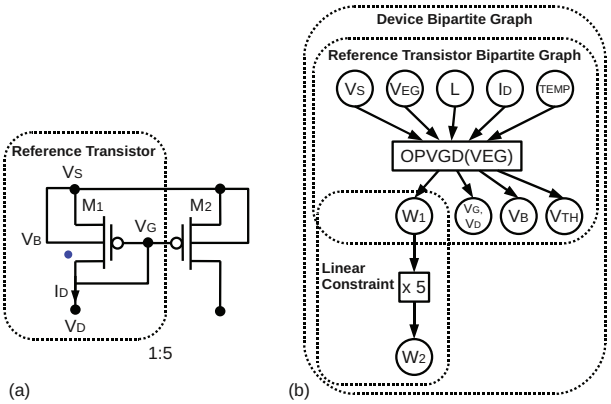

Fig. 5. (a) PMOS current mirror device, (b) its associated bipartite graph.

2) Device Sizing and Biasing: From the bipartite DAG of the reference transistor shown in Fig. 4, more complex bipartite graphs are built for devices. Fig. 5(a) illustrates a current mirror device. The designer can set a current ratio of $1:5$ from transistor M_1 to transistor M_2. Therefore the linear constraint $W_2 = 5 \cdot W_1$ must be respected. The bipartite graph for the current mirror is shown in Fig. 5(b). Operator $OPVGD(V_{EG})$ computes the width W_1 for the reference transistor as in Fig. 4(b), then the designer constraint is added in the graph to compute W_2.

3) Circuit Sizing and Biasing: A bipartite DAG is associated to a circuit. As for an example, the bipartite DAG associated to the folded cascode OTA shown in Fig. 7 is represented in Fig. 8. The designer chooses the P_{in} set of input parameters according to his intent, and propagates their values to the reference transistor of each device using constraints. Next he may declare designer-defined procedures (DDP) to express the knowledge that cannot be automatically extracted. P_{out} contains the widths and biases computed by the sizing operators, after the graph evaluation from top to bottom.

IV. Fixed Point Iteration Implementation

To compute the transistor width, we focus on solving the problem described in equation (1). We need to find the width W^* that satisfies a target current I_{D_T} chosen by the designer. W^* is found by solving the following equation :

$$I_D(W) - I_{D_T} = 0 \tag{5}$$

where $I_D(W)$ is computed from a given transistor compact model. In a first approach, equation (5) was solved using the Newton-Raphson method. However, as stated in section IV-A, the main drawback of this method is the compulsory computation of the derivative $I'_D(W)$. Thus we use the fixed point iteration algorithm to simplify the root solving algorithm implementation into the sizing and biasing operators.

A. Fixed Point Iteration Algorithm Definition

The fixed point iteration [21] is an algorithm that allows to solve nonlinear equations that have the following form :

$$x = g(x), \quad g : [a,b] \rightarrow [a,b] \tag{6}$$

i.e. find the root $x^* \in [a,b]$ that verifies : $x^* = g(x^*)$. The fixed point iteration consists in choosing an initial estimate $x_0 \in [a,b]$, and generating the sequence $\{x_n\}$ recursively, using the relation below :

$$x_n = g(x_{n-1}) \tag{7}$$

where n is the current iteration. The recurrent equation (7) is performed until it converges, under some condition, towards $x^* \in [a,b]$. Thus x^* is called a *fixed point* of the iteration defined by the function g. The convergence condition for the fixed point iteration is given by :

$$|g'(x)| < 1, \forall x \in [a,b] \tag{8}$$

The stopping criteria for the fixed point iteration is given as follows :

$$|x_n - x_{n-1}| < \epsilon \tag{9}$$

where ϵ is usually related to x_0.

Another root-finding approach for nonlinear equations is the Newton-Raphson method, used to solve equations in the form of $f(x) = 0$. The Newton-Raphson iteration is defined by :

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})} \tag{10}$$

As shown in equation (10), the Newton-Raphson iteration requires to compute a derivative f', whereas the fixed point iteration (equation (7)) does not require any derivative computation. Thus, using the fixed point iteration highly simplifies the root-finding algorithm implementation in the sizing and biasing operators, which is an improvement over the Newton-Raphson method. Moreover, according to [22], it is possible to accelerate the fixed point convergence using the *Aitken method*, that consists in computing the $\{x_n, n \geq 2\}$ sequence using :

$$x_{n+1} = \frac{x_{n-2} \cdot x_n - (x_{n-1})^2}{x_n - 2x_{n-1} + x_{n-2}}, n \geq 2 \tag{11}$$

B. Transistor Width Computation with Fixed Point Iteration

To apply the fixed point iteration algorithm on the transistor width computation, we first transform equation (5) into :

$$\frac{I_D(W)}{I_{D_T}} - 1 = 0 \tag{12}$$

Then, using equation (6), we rewrite equation (12) as :

$$W = \frac{I_D(W)}{I_{D_T}} - 1 + W = g(W) \tag{13}$$

The fixed point iteration corresponding to equation (13) is given below :

$$W_n = \frac{I_D(W_{n-1})}{I_{D_T}} - 1 + W_{n-1} = g(W_{n-1}) \tag{14}$$

We choose the initial value W_0 to be :

$$W_0 = 10 \cdot W_{min} \tag{15}$$

where W_{min} is the minimum width for a given technology.

Considering the convergence condition stated in equation (8), the derivative of $g(W)$ in equation (13) is given by :

$$g'(W) = \frac{I'_D(W)}{I_{D_T}} + 1 \tag{16}$$

Assuming that $I_D(W)$ is linear, it may be stated that $\frac{I'_D(W)}{I_{D_T}}$ is positive. Thus $|g'(W)| > 1$ (see Fig. 6), *i.e.* the convergence condition in equation (8) is not respected. Thus the fixed point iteration written in equation (14) will not converge.

Now, using the method described in [22], it is possible to modify equation (6) to make the corresponding iteration converging. Adding αx on both sides of equation (6) gives :

$$x + \alpha x = g(x) + \alpha x \tag{17}$$

Equation (17) leads to :

$$x = g_1(x) \tag{18}$$

where $g_1(x)$ is expressed as follows :

$$g_1(x) = \frac{\alpha x + g(x)}{1 + \alpha},\ \alpha \neq -1 \tag{19}$$

Equation (18) becomes the new equation to be solved with the fixed point iteration. The derivative of $g_1(x)$ is given by :

$$g'_1(x) = \frac{\alpha + g'(x)}{1 + \alpha},\ \alpha \neq -1 \tag{20}$$

Equation (20) shows that α parameter can be tuned to make convergent the fixed point iteration with g_1 function.

Now, applying equation (17) to our case (equation (13)) gives :

$$W + \alpha W = \frac{I_D(W)}{I_{D_T}} - 1 + W + \alpha W \tag{21}$$

Rewriting equation (21) leads to :

$$W = \frac{1}{1 + \alpha}(\frac{I_D(W)}{I_{D_T}} - 1) + W = g_1(W) \tag{22}$$

The new fixed point iteration, corresponding to equation (22), is given below :

$$W_n = \frac{1}{1 + \alpha}(\frac{I_D(W_{n-1})}{I_{D_T}} - 1) + W_{n-1} = g_1(W_{n-1}) \tag{23}$$

This iteration is guaranteed to converge when using the suitable α parameter. Knowing that $g(W)$ is approximately linear implies that $g'(W)$ is a constant. Setting $\alpha = -g'(W)$ in equation (20) (with replacing x by W) gives : $g'_1(W) = 0$, what respects the convergence condition in equation (8). Thus, we compute α as the slope of $g(W)$ using :

$$\alpha = -g'(W) \approx -\frac{g(10 \cdot W_{min}) - g(W_{min})}{10 \cdot W_{min} - W_{min}} \tag{24}$$

α parameter is computed once, it is the same for all iterations. The iteration in equation (23) is implemented within the sizing and biasing operators that are used in the bipartite graph. Moreover, the Aitken acceleration method (equation (11)) is also implemented to decrease the computation time.

Examples of $g(W)$ (equation (13)) and $g_1(W)$ (equation (22)) are represented in Fig. 6, with the same I_{D_T} and transistor biasing. $f(W) = W$ is the function that intersects $g(W)$ and $g_1(W)$ when $W = W^*$. It can be seen on this figure that $|g'(W)|$ is very high (typical values of $|g'(W)|$ are around 10^5), whereas $|g'_1(W)| < 1$.

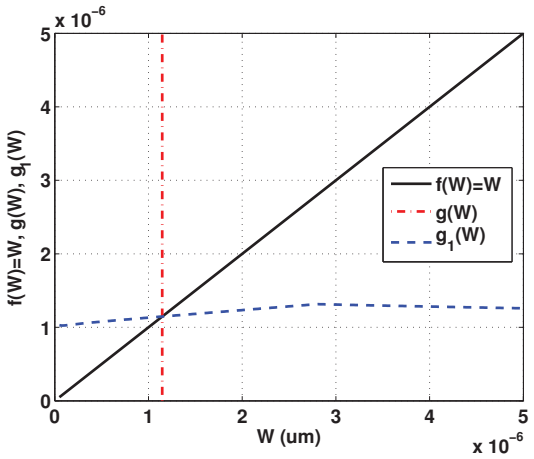

Fig. 6. Representation of $g(W)$ and $g_1(W)$. The solution W^* computed by the fixed point iteration is at the intersection of $f(W) = W$ and $g_1(W)$. Note that $|g'(W)|$ is very high, whereas $|g'_1(W)| < 1$.

V. Case Study

Using the new formulation for analog circuits sizing based on the fixed point iteration algorithm, we design the folded cascode OTA represented in Fig. 7. It is composed of six devices : D1, D1C, D2, D2C, D3 and D4, C_L is the load capacitance. The folded cascode OTA is first designed in a 130nm technology, then it is migrated to a 65nm process. Both design and migration are performed using the same bipartite graph.

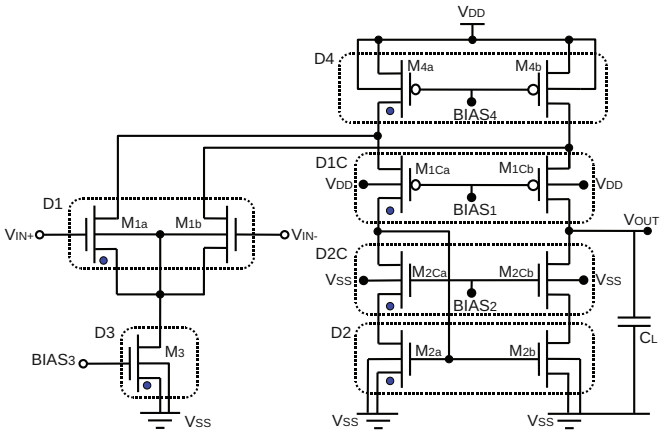

Fig. 7. Folded cascode OTA. Each dashed-box is a device, in which the reference transistor is marked with a dot.

A. Folded Cascode OTA Bipartite Graph

First, the bipartite graph associated to the folded cascode OTA is generated, it is shown in Fig. 8. The bipartite graph represents the sizing procedure for the circuit, its evaluation from top to bottom provides transistors sizes and biases. The designer allocates the input parameters P_{in} (at the top of the graph in Fig. 8), then their value (see Table II) is spread in the graph and used by the sizing and biasing operators to compute unknown sizes and biases. The rectangle nodes named *eq1* and *eq2* represent designer defined equations, they are used

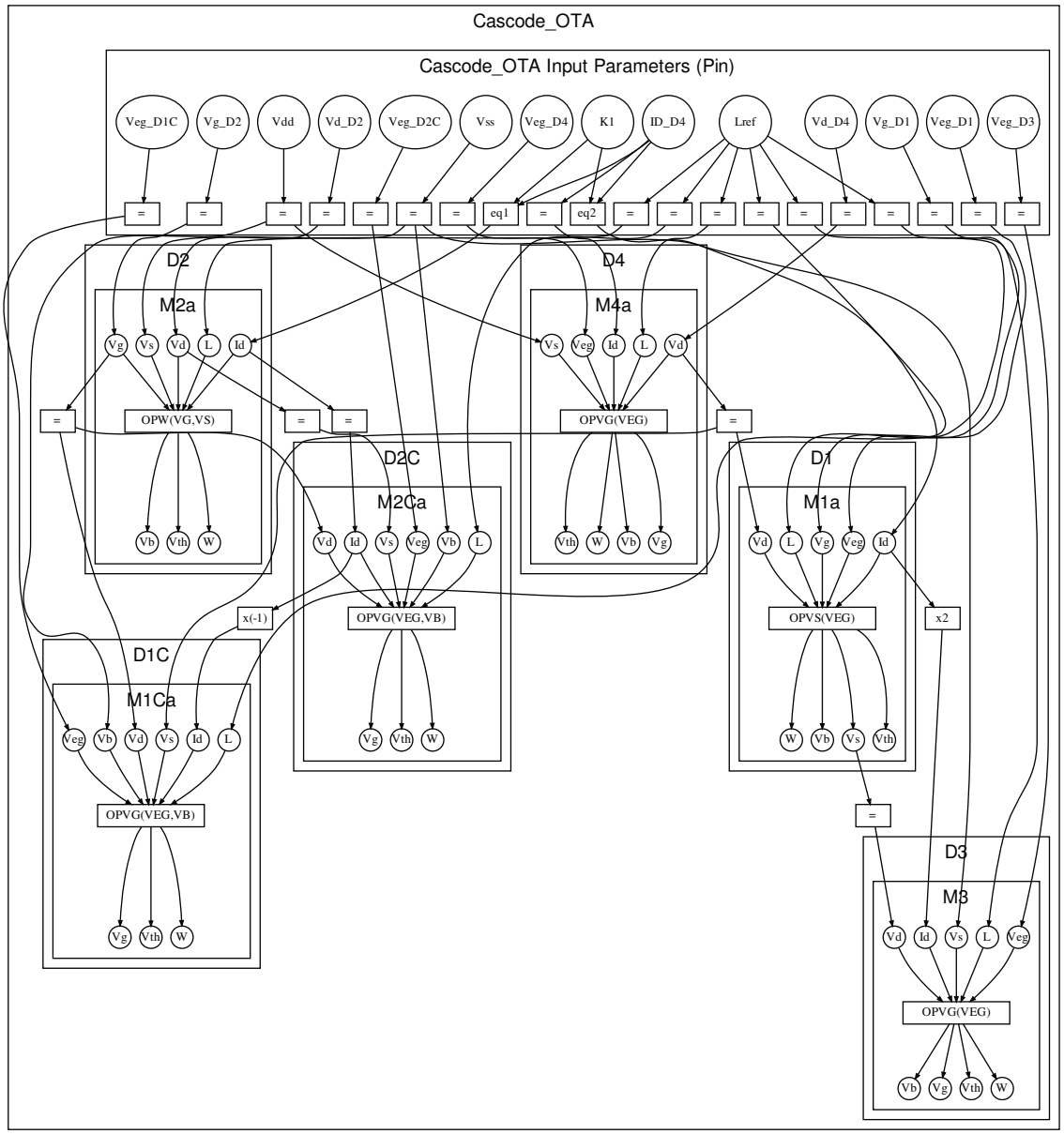

Fig. 8. The bipartite graph (*i.e.* the design procedure) associated to the folded cascode OTA. Sizing and biasing operators are part of the bipartite graph. Input parameters P_{in} (see Table II) are on the top of the graph.

to define current ratios between several transistors, as defined below :

$$\begin{array}{ll} eq1: & I_{D,D2} = K_1 \cdot I_{D,D4} \\ eq2: & I_{D,D1} = |I_{D,D4}| - |K_1 \cdot I_{D,D4}| \end{array} \quad (25)$$

B. Folded Cascode OTA Design in a 130nm Technology

The folded cascode OTA is sized using a 130nm technology (BSIM3v3 model) and the input parameters P_{in} listed in Table II. The reference current $I_{D,D4}$ is defined as the current of transistor M4a. The bipartite graph evaluation provides the widths and voltages that are respectively listed in Tables III and IV. Using these computed results, the folded cascode amplifier is simulated. Simulation results are shown in Fig. 9 : the DC gain is equal to 60.8dB, phase margin is equal to $75.4°$ and the transition frequency is 94.4MHz. The load capacitance C_L is set to 0.5pF.

C. Folded Cascode OTA Migration to a 65nm Technology

The folded cascode OTA is then migrated to a 65nm process (BSIM4 model), using the same bipartite graph (*i.e.* the same design procedure, see Fig. 8). Our goal is to keep the same transition frequency and phase margin. Thus we tune the input parameters P_{in} (Table II) and evaluate again the bipartite graph. $I_{D,D4}$ is increased, current ratio K_1 is modified to

TABLE II
INPUT PARAMETERS P_{in} FOR THE FOLDED CASCODE OTA DESIGN.

Parameter	130nm Sizing	65nm Migration
$Temp$ (Kelvin)	300.15	300.15
V_{DD} (V)	1.2	1.2
V_{SS} (V)	0.0	0.0
$I_{D,D4}$ (μA)	-50	-66.6
$L_{ref}(\mu m)$	0.5	0.5
$V_{EG,D4}$ (V)	-0.12	-0.1
$V_{EG,D1C}$ (V)	-0.12	-0.1
$V_{EG,D2C}$ (V)	0.12	0.1
$V_{EG,D1}$ (V)	0.12	0.0
$V_{EG,D3}$ (V)	0.12	0.05
$V_{D,D4}$ (V)	0.9	0.95
$V_{D,D2}$ (V)	0.3	0.3
$V_{G,D1}$ (V)	0.6	0.6
$V_{G,D2}$ (V)	0.6	0.6
$K_1^{(*)} = I_{D,D2}/I_{D,D4}$	-0.5	-0.25

$^{(*)}$ See equation (25).

TABLE III
COMPUTED WIDTHS FOR THE FOLDED CASCODE OTA.

Device	Computed Width (μm) 130nm Technology	Computed Width (μm) 65nm Technology
D1	4.9	11.6
D1C	17.3	7.4
D2	1.4	1.7
D2C	5.2	2.2
D3	11	17
D4	33.6	22.8

TABLE IV
COMPUTED VOLTAGES FOR THE FOLDED CASCODE OTA.

Device	Computed Voltage (V) 130nm Technology	Computed Voltage (V) 65nm Technology
D1	$V_S = 0.14$	$V_S = 0.12$
D1C	$V_G = 0.37$	$V_G = 0.36$
D2C	$V_G = 0.82$	$V_G = 0.92$
D3	$V_G = 0.46$	$V_G = 0.53$
D4	$V_G = 0.73$	$V_G = 0.64$

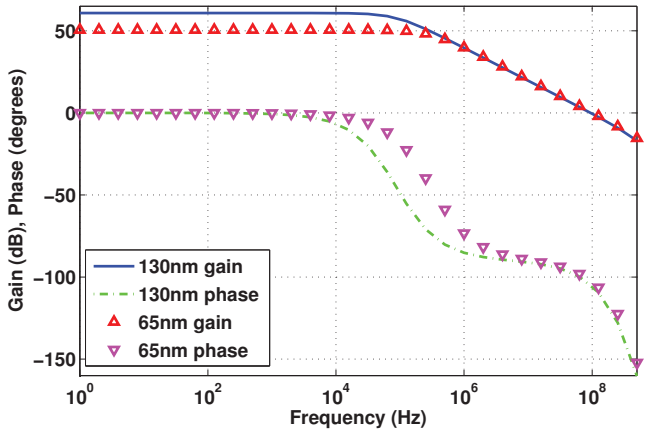

Fig. 9. Simulation results for the folded cascode OTA in 130nm and 65nm technologies.

ensure a suitable DC gain compared to the 130nm technology. $V_{EG,D3}$ is decreased to ensure the operation of M3 transistor in saturation region. $V_{EG,D1}$ is set to 0V, thus M1a and M1b transistors are operating near subthreshold in saturation region. C_L is set to 0.6pF. Simulation results are shown in Fig. 9 : the DC gain is 50.5dB, the phase margin is 76.9°, and the transition frequency is 101MHz. Computed widths and voltages are respectively listed in Tables III and IV. We succeed to maintain transition frequency and phase margin, but the DC gain is lowered. Thus design trade-offs become very important when migrating. The very low evaluation time of the bipartite graph allows to explore design trade-offs in an interactively.

D. Fixed Point Iteration Computational Efficiency

Table V presents the computation time (*i.e.* the evaluation time of the bipartite graph in Fig. 8 from top to bottom) and total number of iterations during the sizing and migration of the folded cascode OTA, for the fixed point iteration and the Newton-Raphson algorithms. The fixed point algorithm takes a few more milliseconds compared to the Newton-Raphson method, this is due to the computation of α parameter for each transistor, that requires two more simulations at the fixed point algorithm initialization. The main advantage of the fixed point iteration algorithm is its efficient and simple implementation into the sizing and biasing operators, what enhances its stability. This is very promising for our future work with nanoscale CMOS technologies [15].

TABLE V
NEWTON-RAPHSON METHOD VS FIXED POINT ITERATION

130nm Technology		
Algorithm	Computation Time (ms)	Total Number of Iterations
Newton-Raphson	16	44
Fixed Point	20	45
65nm Technology		
Algorithm	Computation Time (ms)	Total Number of Iterations
Newton-Raphson	16	42
Fixed Point	21	42

VI. CONCLUSION

An efficient algorithm for analog circuits sizing that uses transistor compact models within the fixed point iteration has been presented. The fixed point iteration algorithm does not require any derivative computation, thus highly simplifies the root solving procedure implemented into the sizing and biasing operators. Moreover it has been demonstrated that the fixed point iteration is guaranteed to converge and that convergence can be highly accelerated. Using sizing and biasing operators implementing the fixed point iteration algorithm, a folded cascode OTA was successfully sized in a 130nm process (BSIM3v3 model), and migrated to a 65nm process (BSIM4 model). Sizing and migration were performed using the same bipartite graph (*i.e.* design procedure). Design trade-offs were rapidly explored thanks to the very low evaluation time of the bipartite graph.

REFERENCES

[1] Ramy Iskander, Marie-Minerve Louërat, and Andreas Kaiser. "Hierarchical Sizing and Biasing of Analog Firm Intellectual Properties". *Integration, the VLSI Journal*, 2012. in press, DOI 10.1016/j.vlsi.2012.01.001.

[2] R. Harjani, R. A. Rutenbar, and L. R. Carley. "OASYS: A Framework for Analog Circuit Synthesis". *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 8(12):1247–1266, December 1989.

[3] M. G. R. DeGrauwe, E. Dijkstra O. Nys, J. Rijmenants, S. Bitz, B. L. A. G. Goffart, E. A. Vittoz, S. Cserveny, C. Meixenberger, G. van der Stappen, and H. J. Oguey. "IDAC: An Interactive Design Tool for Analog CMOS Circuits". *IEEE J. of Solid-State Circuits*, SC-22(6):1106–1116, December 1987.

[4] Han Young Koh, Carlo H. Sequin, and Paul R. Gray. "OPASYN: A Compiler for CMOS Operational Amplifiers". *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 9(2):113–125, February 1990.

[5] W. Nye, D.C. Riley, A. Sangiovanni-Vincentelli, and A. L. Tits. "DELIGHT.SPICE: An Optimization-Based System for Design of Integrated Circuits". *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 7(4):501–518, April 1988.

[6] E. S. Ochotta, R. A. Rutenbar, and L. R. Carley. "Synthesis of High-Performance Analog Circuits in ASTRX/OBLX". *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 15(3):273–294, March 1996.

[7] G. Van der Plas, G. Debyser, F. Leyn, K. Lampaert, J. Vandenbussche, G. Gielen, W. Sansen, P. Veselinovic, and D. Leenaerts. "AMGIE-A synthesis environment for CMOS analog integrated circuits". *IEEE Trans. on Circuits Syst.*, 20(9):1037–1058, September 2001.

[8] M. Krasnicki, R. Phelps, R. A. Rutenbar, and L. R. Carley. "MAELSTROM: Efficient Simulation-based Synthesis for Custom Analog Cells". *Proc. Design Automation Conf.*, pages 945–950, June 1999.

[9] R. Phelps, M. Krasnicki, R. A. Rutenbar, L. R. Carley, and J. R. Hellums. "ANACONDA: Robust Synthesis of Analog Circuits Via Stochastic Pattern Search". *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, pages 567–570, 2000.

[10] Jacky Porte. *"COMDIAC: Compilateur de Dispositifs Actifs"*. Ecole Nationale Supérieure des Télécommunications, Paris, September 1997.

[11] D. Stefanovic, M. Kayal, and M. Pastre. "PAD: A new interactive Knowledge-Based Analog Design Approach". pages 291–299, March 2005.

[12] Danica Stefanovic and Maher Kayal. *"Structured Analog CMOS Design"*. Kluwer Academic Publishers, 2009.

[13] D. M. Binkley, C. E. Hopper, S. D. Tucker, B. C. Moss, J. M. Rochelle, and D. P. Foty. "A CAD Methodology for Optimizing Transistor Current and Sizing in Analog CMOS Design". *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 22(2):225–237, February 2003.

[14] F. Javid, R. Iskander, and M-M. Louërat. "Simulation-Based Hierarchical Sizing and Biasing of Analog Firm IPs". *IEEE International Behavioral Modeling and Simulation Conference*, pages 43–48, September 2009.

[15] L. Lewyn, T. Ytterdal, C. Wulff, and K. Martin. "Analog Circuit Design in Nanoscale CMOS Technologies". *Proceedings of the IEEE*, Vol. 97(No. 10):1687–1714, October 2009.

[16] William Liu. *"MOSFET Models for SPICE Simulation Including BSIM3v3 and BSIM4"*. Wiley-Interscience, 2001.

[17] NXP, MOS Model PSP level 103, 2011.

[18] Christian Enz, François Krummenacher, and Eric Vittoz. "An Analytical MOS Transistor Model Valid in All Regions of Operation and Dedicated to Low-Voltage and Low-Current Applications". *Analog Integrated Circuits and Signal Processing Journal*, Vol. 8(No. 1):83–114, July 1995.

[19] D. Libes. EXPECT. 2010.

[20] F. Javid, R. Iskander, M-M. Louërat, and D. Dupuis. "Analog Circuits Sizing Using Bipartite Graphs". *IEEE Midwest Symposium on Circuits and Systems*, August 2011.

[21] P. Henrici. *"Elements of Numerical Analysis"*. Wiley, John Sons, 1964.

[22] J. Legras. *"Méthodes et Techniques de l'Analyse Numérique"*. Dunod, 1971.

Farakh JAVID received the B.S. and M.S. degrees in computer science in 2006 and 2008 respectively, both from Université Pierre et Marie Curie, Paris, France. In 2008 he joined the LIP6 laboratory to develop a standard interface with simulators for an analog circuits design tool. Since 2009 he is working as Ph.D. candidate at LIP6 laboratory on analog design automation. His research interests are knowledge management and algorithms for analog design automation.

Ramy ISKANDER received the M.S. degree in 2004 and the Ph.D. degree in 2008, from the Laboratoire d'Informatique de Paris 6 (LIP6) at Université Pierre et Marie Curie (UPMC), Paris, France. Before, he worked for more than 10 years in many international EDA companies. Currently, he is an Associate Professor in the AMS group within the Laboratoire LIP6 at the same university. He published more than 40 papers on analog design automation methods for nanometer technologies covering modeling, simulation, synthesis, layout generation and technology migration. He served as a reviewer for French National ANR projects and for several book reviews, international conferences and international journal papers. Currently, he is the scientific coordinator of the European FP7/ICT/Green Car project called AUTOMICS covering modeling and simulation of substrate coupling effects for automotive applications.

François DURBIN received the degree of Ingénieur E.S.E in 1964. From 1964 to 1967 he studied for the MSc. degree in solid state physics (Witwatersrand University, Johannesburg, South Africa). Since 1968 he is affiliated with the French Atomic Energy Commission (CEA), in an Electronics Department, as a research engineer and numerical analyst. His research areas include CAD of large radiation hardened circuits, development of dedicated architectures for signal processing (Wigner-Ville and Kalman), large scale system optimization by generalized simulated annealing. He has been a scientific adviser for 21 Ph.D. theses in the above mentioned fields, and has authored and co-authored about 30 papers in the field of circuit CAD, signal processing, and optimization.

Marie-Minerve LOUËRAT received the M.S. degree in 1983 and the PhD degree in 1986, both from Université d'Orsay (Paris XI). In 1986, she joined the Centre National de la Recherche Scientifique (CNRS) where she is today responsible for the AMS EDA group within the Laboratoire d'Informatique de Paris 6 (LIP6) at Université Pierre et Marie Curie (Paris 6). She published papers on static timing analysis, data-converters and analog design automation. She served several years as University Booth Chair of the Design Automation and Test in Europe conference (DATE).