Automatic Biasing Point Extraction and Design Plan Generation for Analog IPs

Ramy Iskander
Université Pierre et Marie Curie
LIP6 Laboratory
4 Place Jussieu, 75252 Paris, France
Email: ramy.iskander@lip6.fr

Andreas Kaiser
IEMN-ISEN
41 Bld. Vauban, 59046 Lille Cedex, France
Email: andreas.kaiser@isen.fr

Marie-Minerve Louërat
Université Pierre et Marie Curie
LIP6 Laboratory
4 Place Jussieu, 75252 Paris, France
Email: marie-minerve.louerat@lip6.fr

***Abstract*— In this paper, an algorithm for automatic extraction of DC biasing point towards generation of design plans is presented. Initially, the circuit is described as a hierarchy of modules and devices inside our dedicated framework CAIRO+. Electrical information is propagated from higher level modules, to lower level ones, till reaching the device level. During navigation through the hierarchy, a dependency subgraph is generated for each device and module. Each subgraph expresses electrical dependencies by choosing among a set of predefined sizing operators. To obtain a final directed acyclic graph, existing graph directed cycles are detected and removed. The resulting graph represents the complete sizing procedure for the analog IP. The calculated biasing point is compared against operating point simulation. The algorithm is successfully applied to two analog IPs: single-ended two-stages output transconductance amplifier and differential cascode current-mode integrator.**

I. Introduction

Analog synthesis tools can be categorized in two different classes: simulation-based and knowledge-based. A simulation-based synthesis flow, like MAELSTROM [1], involves an optimizer in a closed loop with an analog simulator. Both communicate circularly till an optimal set of values is achieved for circuit parameters and the specifications are met. The simulation-based approach had the golden reputation of using the actual commercial transistor models, hence, sizing accuracy is always ensured. As distinct from the simulation-based approach, a knowledge-based synthesis flow, like OASYS [2], is mainly based on approximate transistor models. In this approach, the designer has to identify all the circuit equations describing the specifications as functions of circuit parameters. The designer must also identify the sizing procedure based on these equations. Once determined, the sizing procedure can be developed using a programming language as C or Matlab. Despite of its tediousness, the knowledge-based approach is more appealing to the designer than the synthesis-based one. Designers may not trust simulation-based synthesis results or may consider it incomplete. To resolve the problem of inaccurate models, knowledge-based synthesis systems started to incorporate accurate models borrowed from analog simulators. Moreover, the selection of circuit parameters in simulation-based synthesis systems in not so obvious. Most systems directly optimise the dimensions of transistors. Many designers tend to use currents and voltages instead of dimensions. In knowledge-based systems, the designer has full control over the choice of parameters.

Our general objective is to develop a language to document designer's expertise and use it later to synthesize and migrate analog IPs. This language makes part of a more general knowledge-based synthesis system that tries to automate design plan generation. This paper addresses the problem of calculating the biasing point using accurate transistor models and extracting sizing procedures.

Section II defines the problem and shows how to derive circuit sizing procedure through circuit topology. Section III presents biasing and sizing results of two analog IPs. Finally, Section IV concludes the paper.

II. Automatic Biasing and Sizing

A. Problem definition

Knowledge-based synthesis requires from the designer full knowledge of the circuit. Expressing circuit specifications as functions of selected device parameters becomes a tedious task. Most specifications can be expressed in terms of small signal parameters, which in turn, are expressed in terms of device dimensions. Tuning dimensions should allow the designer to reach the specifications required. In order to obtain the dimensions from circuit specifications, the designer has to write a complex sizing procedure. The procedure is fully based on the designer's expertise. Each designer may have different guidelines.

$$\underbrace{\begin{bmatrix} \mathrm{Temp} \\ \mathrm{V_{in,i}} \\ \mathrm{V_{eg,i}} \\ \mathrm{I_{b,i}} \\ \mathrm{V_{out,i}} \\ \mathrm{L_i} \end{bmatrix}}_{\text{Selected Parameters}} \Rightarrow \underbrace{\begin{bmatrix} \mathrm{W_1} \\ \mathrm{W_2} \\ \mathrm{W_3} \\ . \\ . \\ \mathrm{W_n} \end{bmatrix}}_{\text{Device Dimensions}} \Rightarrow \underbrace{\begin{bmatrix} \mathrm{gm_1} \\ \mathrm{gds_1} \\ \mathrm{C_{gs1}} \\ . \\ . \\ \mathrm{gm_n} \end{bmatrix}}_{\text{Small – Signal Parameters}} \Rightarrow \underbrace{\begin{bmatrix} \mathrm{A_d} \\ \phi_\mathrm{m} \\ \mathrm{F_t} \\ . \\ . \\ \mathrm{PSRR} \end{bmatrix}}_{\text{Circuit Specifications}}$$

Fig. 1. Mappings in the design space.

B. Objectives

Our aim is to automate the mapping process starting from the selected circuit parameters till reaching the small signal parameters. Starting with primary selected circuit parameters, the automation should address the first three steps of the whole mapping process illustrated in Fig. 1.

C. Biasing point extraction

To calculate the width of a transistor (Fig. 1), we need to determine a priori the following quantities: $temp$, I_{ds}, L, V_{eg} or V_{gs}, V_{ds}, V_{bs}. Normally, $temp$, I_{ds}, L, V_{eg} are fixed by the designer and V_{bs} by the circuit topology. As $V_{eg} = V_{gs} - V_{th}$, V_{eg} fixes V_{gs} when an estimation for V_{th} is available. The only parameter that should be known a priori or should be fixed through the topology connections is the V_{ds}.

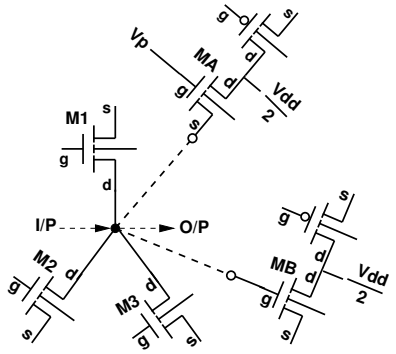

Fig. 2. Possible drain connectivity.

Based on the above observation, an arbitrary analog circuit can be viewed as a set of connections fixing the drain potentials of all the transistors. Actually, very few possibilities are available to fix the drain potential. Let us assume that an arbitrary circuit consists of transistors M_1, M_2 and M_3 as in Fig. 2 . To size the three transistors, the potential of the common drain node should be fixed. The following are the only possible situations:

1) The drain terminal may be considered as an input voltage. It should be fixed by the designer a priori
2) The drain terminal may be considered as an output voltage. It should be fixed by the designer a priori
3) The drain terminal may be connected to the source of transistor M_A. It becomes an unknown
4) The drain terminal may be connected to the gate of transistor M_B. It becomes an unknown

Points 3 and 4 could be resolved by inverting the BSIM3V3 model. In other words, we will inverse numerically the BSIM3V3 analytical model to calculate V_s, V_g, V_{th} and W as a function of V_d. The study of the inversion of the BSIM3V3 transistor model, resulted in developing the analogical sizing operators described in subsequent sections.

D. Hierarchical synthesis in CAIRO+ framework

Analog circuits can be described in CAIRO+ framework [3] as a hierarchy of interconnected *devices* and *modules*. Higher-level modules instantiate lower-level modules and devices. Devices and modules are available as libraries.

During instantiation, each device declares a complete synthesis procedure. In the synthesis procedure, the device declares electrical constraints and make a call to the synthesis kernel. For modules, the synthesis procedure passes the values of all known electrical quantities to child modules or devices and then makes a call to the synthesis kernel. The synthesis process is hierarchical in the sense that each device and each module have no access to a higher scope. Each device and module only encapsulates its own electrical dependencies.

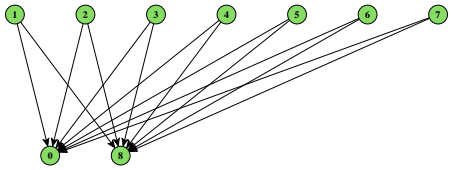

Fig. 3. Dependency subgraph of a transistor.

TABLE I
NODE LEGENDS FOR A TRANSISTOR

Index	Label	Index	Label
0	V_g	5	V_d
1	$Temp$	6	V_b
2	I_{ds}	7	V_s
3	L	8	W
4	V_{eg}		

The result of the synthesis process is a directed acyclic graph representing all the electrical dependencies between all electrical parameters of the devices. In this graph, a node represents one electrical device parameter and an edge $v \leftarrow u$ represents the dependency of v on u. Fig. 3 shows an example of a dependency subgraph for a transistor. Table I gives the meaning of each node.

E. Sizing operators

Based on the principal ideas presented in sub-sections C and D, some analogical sizing operators have been proposed in this work. Each operator can be described mathematically as follows:

$$OPxxx(RVALUE_i, ...) : \quad (1)$$
$$(LVALUE_i, ...) \Leftarrow (RVALUE_{given}, RVALUE_i, ...)$$

where $OPxxx$ is the name of the operator. $LVALUE_i$ are the unknown electrical quantities that are calculated by applying this operator to the device. $RVALUE_{given}$ are the electrical quantities given by the designer. $RVALUE_i$ are the known electrical quantities fixed a priori by the designer that dictates which version of the operator to apply. Operators are often called *dependency rules* or *constraints*. Table II summarises the list of operators that could be applied to devices.

During navigation through the hierarchy, a set of dependency rules is generated for each device. The dependency rule is equivalent to one pre-defined sizing operator. As an example, Fig. 3 shows the dependency rules $OPVG(V_{eg})$ and $OPW(V_{eg})$ under the assumption that V_g and W are

TABLE II

DEFINITION OF SIZING OPERATORS

Operator	Definition
$OPVS(V_{eg})$	$(V_s,V_{th},W) \Leftarrow temp,I_{ds},L,V_{eg},V_d,V_b,V_g$
$OPVS(V_{gs})$	$(V_s,V_{th},W) \Leftarrow temp,I_{ds},L,V_{gs},V_d,V_b,V_g$
$OPVS(V_{eg},W)$	$(V_s,V_{th}) \Leftarrow temp,I_{ds},W,L,V_{eg},V_d,V_b,V_g$
$OPVS(V_{gs},W)$	$(V_s,V_{th}) \Leftarrow temp,I_{ds},W,L,V_{gs},V_d,V_b,V_g$
$OPVG(V_{eg})$	$(V_g,V_{th},W) \Leftarrow temp,I_{ds},L,V_{eg},V_d,V_b,V_s$
$OPVG(V_{gs})$	$(V_g,V_{th},W) \Leftarrow temp,I_{ds},L,V_{gs},V_d,V_b,V_s$
$OPVG(V_{eg},W)$	$(V_g,V_{th}) \Leftarrow temp,I_{ds},W,L,V_{eg},V_d,V_b,V_s$
$OPVG(V_{gs},W)$	$(V_g,V_{th}) \Leftarrow temp,I_{ds},W,L,V_{gs},V_d,V_b,V_s$
$OPVGD(V_{eg})$	$(V_g,V_d,V_{th},W) \Leftarrow temp,I_{ds},L,V_{eg},V_b,V_s$
$OPVGD(V_{gs})$	$(V_g,V_d,V_{th},W) \Leftarrow temp,I_{ds},L,V_{gs},V_b,V_s$
$OPVGD(V_{eg},W)$	$(V_g,V_d,V_{th}) \Leftarrow temp,I_{ds},W,L,V_{eg},V_b,V_s$
$OPVGD(V_{gs},W)$	$(V_g,V_d,V_{th}) \Leftarrow temp,I_{ds},W,L,V_{gs},V_b,V_s$
$OPIDS(V_{eg})$	$(I_{ds},V_{th}) \Leftarrow temp,W,L,V_{eg},V_d,V_g,V_b,V_s$
$OPIDS(V_{gs})$	$(I_{ds},V_{th}) \Leftarrow temp,W,L,V_{gs},V_d,V_g,V_b,V_s$
$OPW(V_{eg})$	$(W,V_{th}) \Leftarrow temp,I_{ds},L,V_{eg},V_d,V_g,V_b,V_s$
$OPW(V_{gs})$	$(W,V_{th}) \Leftarrow temp,I_{ds},L,V_{gs},V_d,V_g,V_b,V_s$

unknown. Then, for each device, the dependency rules are merged to form the device subgraph. The subgraphs are then merged to form the module dependency graph. In this graph, nodes represent electrical quantities and edges represent the dependency rules. The process continues until dependencies of all modules and devices are presented in the final graph.

F. Synthesis kernel

Every device and module is considered as a generator. Each generator makes one call to the synthesis kernel in order to create its own dependencies. The synthesis kernel implements the algorithm shown in Fig. 4:

```
function synthesize( generator )
    for all children of the generator
        call synthesize(child)
    end for

    if generator is a device
        generate dependencies for the device
        eliminate all redundant dependencies
    else if generator is a module
        create equipotentials
        merge dependencies of all children generators
        eliminate all redundant dependencies
    end if
end function
```

Fig. 4. Outline of the synthesis function.

The algorithm performs a depth-first traversal by calling recursively the synthesis kernel for all children generators. The process continues until the root generator is synthesized. The root generator is the uppermost module or circuit.

III. SIZING RESULTS

A. Single-ended two-stages OTA

The synthesis algorithm was applied to the OTA amplifier shown in Fig. 5. First, the amplifier was described as a module

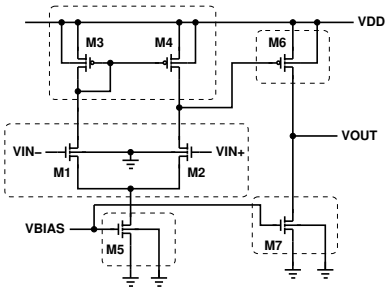

Fig. 5. Single-ended two-stages output transconductance amplifier.

instantiating a hierarchy of devices available as libraries in CAIRO+. The different devices used are marked as dashed boxes. Then, the synthesis procedure of the amplifier was called in order to obtain the transistor dimensions and the small signal parameters. The amplifier was successfully sized and biased.

1) Device dependency subgraph: During synthesis, the dependency subgraph for each device was automatically generated. Fig. 6 shows the dependency subgraph for the current mirror consisting of M_3 and M_4. Table III clarifies the meaning of each node in the subgraph.

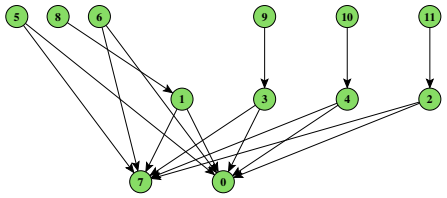

Fig. 6. Dependency subgraph of current mirror consisting of M_3 and M_4.

TABLE III

NODE LEGENDS FOR CURRENT MIRROR SUBGRAPH

Index	Label	Index	Label
0	$V_{g,M_3} = V_{d,M_3} = V_{d_1,cm}$	6	$V_{s,M_3} = V_{s,cm}$
1	$Temp_{M_3} = Temp_{M_4}$	7	$W_{M_3} = W_{M_4}$
2	$I_{ds,M_3} = I_{ds,M_4}$	8	$Temp_{cm}$
3	$L_{M_3} = L_{M_4}$	9	L_{cm}
4	$V_{eg,M_3} = V_{eg,M_4}$	10	$V_{eg,cm}$
5	$V_{b,M_3} = V_{b,cm}$	11	$I_{ref,cm}$

2) Module dependency graph: To obtain the dependency graph of the OTA amplifier, all the device subgraphs are automatically merged. The final graph of the whole OTA amplifier is shown in Fig. 7. As illustrated, the graph is a *directed acyclic graph*, or DAG . This means that the sizing procedure evolves in a top-down approach. Later, the graph could be converted into a C/C++ CAIRO+ code and compiled into an executable generator.

3) Algorithm versus simulation: Executing the parameterised generator results in all the sizes and biasing voltages for all transistors in the circuit. Table IV show the results of sizing and biasing of the current mirror. The algorithm is compared against the operating point simulation using ELDO. The results shows that knowledge-based synthesis systems can attain a precision comparable to an analog simulator by using accurate BSIM3V3 transistor model.

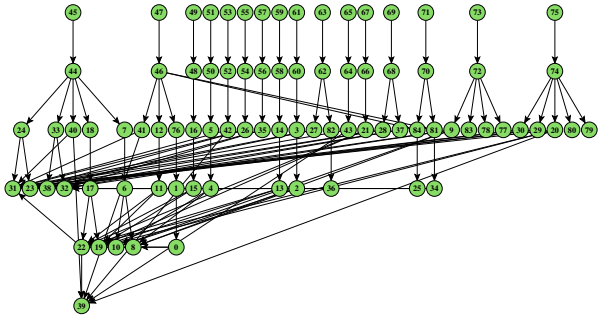

Fig. 7. Dependency graph of the OTA amplifier.

TABLE IV
ALGORITHM VERSUS SIMULATION RESULTS OF THE CURRENT MIRROR

Transistor Parameters	M_3 and M_4 Algorithm	M_3 Simulation	M_4 Simulation
$I_{ds}(\mu A)$	-50.0	-50.02	-50.014
$V_{gs}(V)$	-0.442512	-0.44253	-0.44253
$V_{ds}(V)$	-0.442512	-0.44445	-0.44253
$V_{bs}(V)$	0.0	0.0	0.0
$V_{th}(V)$	-0.342511	-0.34251	-0.34251
$V_{dsat}(V)$	-0.107662	-0.10767	-0.10767
$g_m(fF)(m\Omega^{-1})$	0.72093	0.72115	0.72107
$g_{ds}(\mu\Omega^{-1})$	3.02043	3.0085	3.0211
$g_{mb}(m\Omega^{-1})$	0.158339	0.15839	0.15837
$C_{gd}(fF)$	39.8595	39.838	39.862
$C_{gs}(pF)$	0.657911	0.65788	0.65789
$C_{gb}(pF)$	0.121221	0.12131	0.12131
$C_{sd}(fF)$	0.182819	0.18032	0.18284
$C_{bd}(fF)$	0.354848	0.35009	0.35488

B. Differential cascode current-mode integrator

As a second example, the differential cascode current-mode integrator [4] shown in Fig. 8 was synthesized using CAIRO+. This example illustrates how incomplete knowledge affects the generated dependency graph. Graph anomalies can be easily detected. The detection of such anomalies can help direct the designer to make an optimum choice, during the selection of the minimum set of circuit parameters. The parameters should be fixed a priori in order to synthesize the circuit. As an example, the problem of *graph directed cycles* will be described briefly.

1) Graph directed cycles resolution: Fig. 9 shows a portion of the final generated dependency graph of the integrator. The most interesting aspect is the formation of directed cycles in the graph. Namely, a pair of directed cycles can be identified: $\{(V_{bc},5),(V_{s,m3},9)\}$ and $\{(V_{cp},2),(V_{s,m5},3)\}$. Actually, the appearance of one directed cycle is equivalent to the existence of one unknown that should be specified by the designer prior to the execution of generator. Once specified, it is guaranteed to synthesize the circuit. The resolution of directed cycles is essential as it converts the sizing procedure into a directed acyclic graph. To resolve the directed cycles, the generator warns the designer to specify one unknown from each set of unknowns forming one directed cycle. As an example, one designer can choose to set both V_{bc} and V_{cp}.

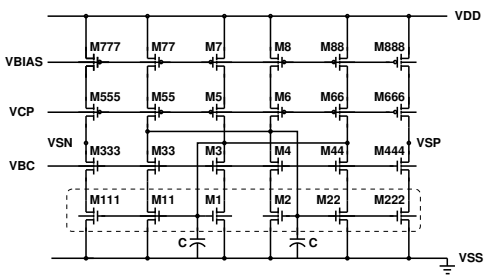

Fig. 8. Differential cascode current-mode integrator.

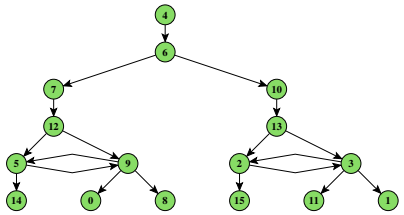

Fig. 9. Portion of the dependency graph of the integrator.

After resolving the graph directed cycles, the minimum total number of circuit parameters that the designer should specify equals the sum of selected circuit parameters and the unknowns resolving detected directed cycles. The whole set of parameters always appears at the root level of the final graph. The final choice of parameters for the integrator agrees with [5].

IV. CONCLUSION

In this paper, an algorithm for automatic extraction of biasing point and generation of suitable sizing procedure was presented. The algorithm was implemented as part of our dedicated analog framework CAIRO+. It was successfully used to synthesize two analog IPs: the single-ended two-stages output transconductance amplifier and differential cascode current-mode integrator. The synthesis in CAIRO+ is purely hierarchical. The simulation results of the OTA amplifier in a 0.13μ technology showed that knowledge-based synthesis could be as accurate as simulation-based synthesis. The synthesis results of the integrator showed the ability of the algorithm to detect designer's incompletely specified knowledge.

REFERENCES

[1] M. Krasnicki, R. Phelps, R. A. Rutenbar and L. R. Carley, *MAELSTROM: Efficient Simulation-based Synthesis for Custom Analog Cells*, Proceedings of Design Automation Conference, pp. 945-950, June 1999.

[2] R. Harjani, R. A. Rutenbar and L. R. Carley, *OASYS: A Framework for Analog Circuit Synthesis*, IEEE Transactions on Computer–Aided Design, vol. 8, No. 12, pp. 1247–1266, Dec. 1989.

[3] P. Nguyen Tuong, M. M. Louërat and A. Greiner, *Guidelines for Designing Smart and Reusable Analog IP Cores*, SAME Sophia Antipolis Microeletronics Forum, Oct. 2004.

[4] S. Smith and E. Sanchez-Sinencio, *Low Voltage Integrators for High-Frequency CMOS Filters Using Current Mode Techniques*, IEEE Transactions on Circuits and Systems - II, vol. 43, No. 1, pp. 39–48, jan 1996.

[5] H. Aboushady and M. M. Louërat, *Low-Power Design of Low-Voltage Current-Mode Integrators for Continuous-Time $\Sigma\Delta$ Modulators*, IEEE International Symposium on Circuits And Systems, pp. 276–279, may 2001.