An Efficient DC Root Solving Algorithm with Guaranteed Convergence for Analog Integrated CMOS Circuits

Francky Leyn, Georges Gielen*, Willy Sansen
Katholieke Universiteit Leuven, Belgium

Abstract

This paper describes a new DC modeling methodology applicable to CMOS integrated circuits. It is named *operating point driven DC formulation* because the operating point is specified directly, and the device dimensions W and L are determined out of it. With other methods, one specifies the device dimensions W and L and determines the operating point. Our method is important for manual design because it allows the designer to reason in terms of voltages and currents and releaves him from the burden of determining device sizes. The algorithm is guaranteed to converge, and is computationally efficient, which allows interactive design space exploration using optimization-based sizing. A design plan used in optimization-based sizing consists for the largest part out of solving the DC part. Speeding up the DC part with a computationally efficient algorithm, that allows parallellisation, results in a boost of optimization speed.

I. Introduction

DC modeling is a topic that received much attention during the past decades. This was mainly due to convergence problems which hampered widespread acceptance of SPICE like simulators. Even today's advanced algorithms sometimes fail to find a DC consistent solution. Also, SPICE DC solving algorithms are targeted towards analysis. The designer, reasoning in voltages and currents, has to specify the device dimensions W and L, leaving him with the burden to determine the latter. In this paper, an algorithm applicable to CMOS circuits is presented that takes voltages and currents as input, converges by construction and is computationally efficient, which is required for optimization-based sizing. In section 2, the application of a DC formulation in an optimization based sizing scheme is discussed. In section 3, the different DC modeling methodologies are discussed and compared. Section 4 introduces the influence of series resistors and junction diodes while section 5 discusses dimension reduction techniques. In section 6, an example is given. Conclusions are drawn in section 7.

II. DC root solving in relation with circuit optimization

A design plan used for optimization-based analog circuit sizing has a structure as depicted in Fig. 1. An executable model is part of an optimization loop. Optimization allows automatisation because it doesn't require explicit understanding of the complex nonlinear relationships between the design parameters. Its drawback is that it requires models for DC, AC and transient behavior that are valid over the part of the design space that fullfills the design requirements. The executable model can be divided in three parts. First the DC part of the device models takes part in a DC root solving loop. After solving the DC equations, the small-signal parameters of the devices are determined using the AC part of their device models. Given the small-signal parameters, one can solve

* Research associate of the Belgian Fund of Scientific Research

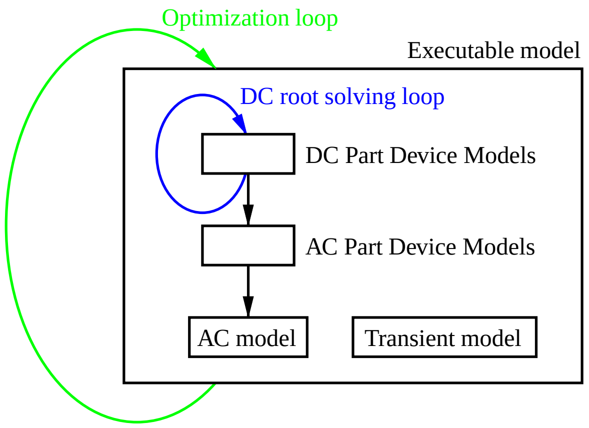

Fig. 1. **Nesting of the DC root solving loop inside the optimization loop.**

the AC model. This can be an *MNA* model, a behavioral signal path model [1], [2] or an *AWE* model [3]. For transient analysis, one can use approximate manual derived expressions or a simulation-based method. In this paper, we concentrate on the DC root solving part since this is the most time consuming part, which is speeded up by the proposed approach.

III. DC modeling

DC modeling can be done in several ways. The method presented here is applicable for sizing of integrated CMOS circuits. We first discuss the different methods and then make a comparison between them. We consider a MOS circuit with N nodes and M MOS transistors.

A. Simulation

In a simulator like SPICE, the DC solution is obtained by solving a system of simultaneous nonlinear equations with a Newton-Raphson root solver. The independent variables are the node voltages $V_n = [v_1, \ldots, v_N]^T$. The number of independent variables of the root solver is thus N. Given the node voltages V_n, one can calculate the branch voltages $V_b = A^T V_n$, with A the incidence matrix [4]. The branch constitutive equations (*BCE*'s) allow to determine the device currents out of the branch voltages and the device dimensions. The constraints are the *KCL*'s of the different nodes. Given the branch currents, one can calculate the error on the *KCL*'s. The system of simultaneous nonlinear equations the root solver has to solve is:

$$KCL(AI_b) = 0, \qquad I_b = BCE(V_b), \qquad V_b = A^T V_n$$

$$\delta_{KCL_i}(V_n) = KCL(A(BCE(A^T V_n))) = 0 \quad i = 1 \ldots N$$

Root solving is an *incomplete* solution method: one can't garantee that the solution will always be found. The iterative Newton-

Raphson scheme can fail to converge. A good measure for the error on the *KCL*'s is the logarithmic difference between the sum of currents that enter and leave a node in function of $\log V_n$. Whether a branch current will flow to or from a node can be determined by inspection of the transistor terminals the branch current originates from.

$$\delta I_{KCL_i}(\log V_n) = \log\left(\sum_{I_{to}\rightarrow\bullet} I_{to}\right) - \log\left(\sum_{\bullet\rightarrow I_{from}} I_{from}\right)$$

In simulation-based optimization, this root solving process is repeated for each iteration in the optimization. For each iteration, the exact DC solution is determined. The inputs of the optimizer are the device dimensions W and L of each transistor and some biasing sources. If the optimizer allows integer variables, on-grid device sizes are obtained.

Obtaining a starting point with a simulation-based DC formulation is problematic. One method is to pass this task to a designer, which excludes automation of the analog sizing flow. An automated but computationally expensive method is generating (W, L) pairs in a random fashion and submitting them to a simulator until the latter converges.

The number of circuit evaluations during an optimization is thus $Opt(M)Root(N)$, with $Opt(n)$ and $Root(n)$ the number of evaluations required to optimize respectively solve a system of order n.

B. Relaxed DC

In a relaxed DC optimization scheme [5], the inner DC root solving loop is merged into the outer optimization loop. Both problems are solved simultaneously. The exact DC solution is not determined for each iteration in the optimization. It is enforced gradually during the optimization and is obtained at the end of the optimization.

A fundamental drawback is that the inner loop is solved with optimization while it should be solved by root solving, which is mathematically a totally different problem. Reformulating a root solving problem into optimization introduces local optima [6, pp. 286–289].

An example can give some flavor on this. Suppose a two-node circuit of which the errors on the *KCL*'s are given by $\delta_{KCL_1}(v_1, v_2) = 5/v_1 + 0.1v_1 - v_2$ and $\delta_{KCL_2}(v_1, v_2) = 0.1v_1^2 - 1.4v_1 + 7 - v_2$. These are both smooth functions of which the unique solution is the intersection of the two loci corresponding to $\delta_{KCL_i} = 0$, as depicted in Fig. 2. These functions are now combined for optimization into the function $\Delta_{KCL} = \delta^2_{KCL_1}(v_1, v_2) + \delta^2_{KCL_2}(v_1, v_2)$. This function is non-convex and has local minima as depicted in Fig. 2. It appears that solving the DC root solving problem with optimization scatters the design space with local optima.

This has two major drawbacks. With a design space scattered full of local minima [3], efficient local optimization [1] is excluded. One is obligated to do the optimization with a computionally expensive global optimizer such as simulated annealing. The possible speed gain obtained by not persuing an exact DC solution during each optimization iteration is lost by this.

A second drawback of the introduced local optima is that DC consistency is less garanteed. DC consistency requires that one achieves the real global optimum. This requires long annealing times. Avoiding DC inconsistency by increasing the weights on the *KCL* error measures is tempting but results in a premature freezing

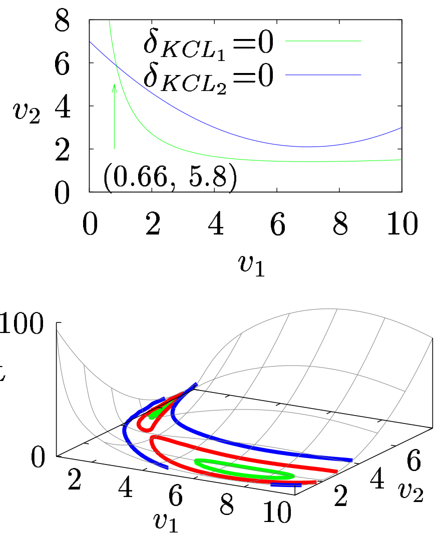

Fig. 2. (a) $\delta KCL_i = 0$ **loci. (b) Surface plot of** ΔKCL

of the node voltages V_n which are key design parameters which should be frozen only at the latest moment. As a consequence, the situation may occur frequently that optimizations have to be run multiple times because the obtained results are DC inconsistent.

In a relaxed DC approach, the node voltages V_n and the device dimensions W and L of the MOS transistors are the independent variables. Therefore, generating an initial point with this method is straightforward. If the optimizer allows integer variables, on-grid device sizes are obtained. For on-grid device sizes, mixed integer nonlinear programming (MINLP) is required. The integer part is required for the device dimensions W and L, the noninteger (real) part for the node voltages V_n. Together with the *BCE*'s, they allow to evaluate the current through all devices and thus the error on all *KCL*'s. The errors on the *KCL*'s are added as penalty terms to the goal function which expresses the design objectives. The number of circuit evaluations is thus $Opt(M + N)$.

C. Operating Point Driven

In *operating point driven sizing*, one specifies the operating point and determines the device dimensions W out of it. The independent variables are the node voltages V_n , a set of independent chord currents I_c and the L's. Out of the node voltages V_n, the branch voltages $V_b = A^T V_n$ are determined, and out of the chosen set of independent chord currents, the branch currents I_b are determined as $I_b = B^T I_c$ with B the basic loopset matrix [4]. Given the L's of the devices, the W's of the devices can be determined since the I_{DS} currents are branch currents that are calculated out of the chord currents:

$$I_b = B^T I_c \qquad B : \textit{Basic loopset matrix}$$

$$\delta_{W_i}(W_i) = I_{DSb} - I_{DSi}(V_{GSi}, V_{DSi}, V_{BSi}, W_i, L_i) = 0 \quad i = 1 \ldots M$$

This is a sequence of M one-dimensional problems. For each transistor, one solves $W = W(I_{DS}, V_{GS}, V_{DS}, V_{BS}, L)$. The root solving is always converging since the function $W(I_{DS}, V_{GS}, V_{DS}, V_{BS}, L)$ is monotonic and thus can be solved with for example a bisection method. A solution is thus guaranteed (even if it is unfeasible $\equiv W < W_{MIN}$), making it a *complete* method. The root solving in W of $\delta_W(W) = I_{DS} - I_{DS}(W) =$

$I_{DS} - I_{DS}(V_{GS}, V_{DS}, V_{BS}, \underline{W}, L) = 0$ can be accelerated by exploiting the known function form of $\delta_W(W)$. Level-1 models don't require root solving since an explicit equation for $W()$ exist. The level-1 W value is a good starting point for higher-order models.

Once the branch currents I_b and the branch voltages V_b are known, the root solving in W for each transistor individually can start. As for the other methods, this allows parallellisation at the level of the device equations. This can drastically boost DC root solving speed, since the DC equations comprise the largest part of a design plan. Given a processor bus connecting multiple processors, the Pthreads multithreading POSIX standard [7] can be used to spread the M root solving processes $\delta_{W_i}(W_i) = 0,\ i = 1, \ldots, M$ over the multiple processors.

Since the W of a transistor is the result of a root solving process, the obtained W's most likely will be off-grid. A snap to grid method is required to obtain on-grid W's. The most appropriate way to do so, is by using a simulation-based DC formulation. A fast approximate Hermite interpolating model can be used to exhaustively try on-grid solutions in the neighborhood of the original solution which fulfill the specifications. The risk of divergence is absent since one is in the close neighbourhood of a known DC consistent operating point.

The generation of a staring point with an operating point driven DC formulation is straightforward since one specifies the operating point directly. The only condition is that all transistor must be on. Neglecting the bulk effect on the threshold voltage by setting $V_T = V_{T0} + \epsilon$, allows to write a set of inequality functions which are linear in the node voltages: $h_i = -V_{DSi}, g_i = -(V_{GSi} - V_{Ti}), i = 1 \ldots M$. Solving the minimax problem:

$$\varphi(V_n^*) = \min_{V_n \in \mathbb{R}} \max_i (h_i(V_n), g_i(V_n))$$

delivers a starting point where all transistors are on. Another method is generating node voltages at random until one obtains one fulfilling all inequality functions. Both methods can also be used for a relaxed DC formulation.

Scaling of the independent input variables, performance variables and design requirements is essential in order to obtain good conditioned optimization problems. The purpose of *scaling* is to linearize strongly nonlinear functions to a maximum extent, in order to ease and accelerate the optimization process. Almost all encountered equations in analog circuit design have logarithmic behavior. For the device equations, $\log I_{DS}$ depends in an asymptotically linear way on $\log V_{GS}$ and $\log V_{DS}$. The same is valid for the small-signal parameters. The expressions for poles also have a logarithmic nature (bode/pole-zero plot). Logarithmic transformations are therefore the appropriate scaling method. The root solving equations are written in $\log W_i$, while the optimization variables are $\log V_n$:

$$\delta_{W_i}(\log W_i) = \log I_{DSb} - \log I_{DSi}(log V_n, \log W_i) = 0 \quad i = 1 \ldots M$$

D. Comparison

In table I, the properties of the previous discussed methods are summarised. Since *Opt* and *Root* have at best quadratic convergence properties, the fastest method is the one with lowest order. Multiple arguments are in favor of operating point driven sizing concerning the required number of circuit evaluations. First, the effort required for root solving is significantly lower than for the other methods. In general, it is more efficient to solve a set of M one-dimensional problems than one big problem of order N. The optimization and root solving part are separated, which avoids a large order for optimization as in the case of relaxed DC. Convergence and consistency problems are absent. On-grid device sizes can be obtained with a snap to grid method.

	DC problems	Sizing order
Simulation	Convergency	Opt(2M) Root(N)
Relaxed DC	Consistency	Opt(2M+N)
Operating Point Driven	N ☺	Opt(N+#I_c+M) M Root(1)

TABLE I

Comparison between different DC modeling methods concerning potential problems and sizing order.

IV. Series resistors and junction diodes

The above reasoning for operating point driven DC formulation neglected the presence of series resistors and junction diodes in order not to complicate the reasoning. They are now taken into account (see Fig. 3). Suppose only junction diodes are present. If these diodes are not shorted (B-S), they are connected between a node and a power supply rail. Since the node voltages are known, their leakage current can be determined. Given the set of independ chord currents, the branch currents can be determined using an adapted basic loopset matrix incorporating equivalent leakage current sources, allowing the method to be applied as before.

Consider the case where only series resistors are present. Although the current through these devices is known, the voltage across their terminals can't be determined since their resistance value depends on the W of the device. However, the W can only be determined if the internal voltages V_{GSi}, V_{DSi}, and V_{BSi} are known, something which is only the case if the series resistors are known. This makes that the device equations of the internal transistor and the series resistors must be solved simultaneously. This can be solved by applying the root solving equations to device models including series resistors.

If both series resistors and junction diodes are present, the situation is more difficult. The series resistors require the device model to be solved as a whole, making that the node voltages of the internal device nodes D_i and S_i, and thus the diode leakage currents, can only be determined afterwards. However, the diode leakage was a prerequisite to determine the branch currents which were required to solve the DC root solving equations. This simultanity can be broken by reconnecting the non-shorted junction diodes from the internal nodes D_i or S_i to the external nodes D respectively S. This approximation introduces a negligible DC error, which should have no influence on the manufacturability of

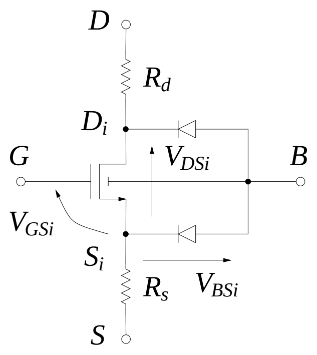

Fig. 3. **Series resisistors and junction diodes of a MOS device.**

the circuit to start with. For the AC model, the normal connectivity pattern can be used.

V. Dimension reduction techniques

The number of optimization variables can be reduced by the methods below which are applicable to all DC modeling methods.

A. Minimal area devices

The effective gate area of some MOS transistors has a dominant influence on performance parameters like offset voltage and noise. If specifications are given for these performance parameters, these transistors may not be designed as minimal area devices. The remaining transistors however can and should be designed towards minimal area. A tempting way to do so is to fix their L to $L = L_{MIN}$. However, since minimal area can correspond to both $L = L_{MIN}$ or $W = W_{MIN}$, setting $L = L_{MIN}$ is equivalent to cutting away a part of the design space. In order to avoid this, the ratio variable $r = W/L$ is used. The turnover ratio r_{MIN} is defined as $r_{MIN} = W_{MIN}/L_{MIN}$, which is higher than one for deep submicron processes. The device geometry and root solving equations become:

$$r \leq r_{MIN} \rightarrow \begin{cases} W = W_{MIN} \\ L = W/r \end{cases} \qquad r > r_{MIN} \rightarrow \begin{cases} L = L_{MIN} \\ W = rL \end{cases}$$

$$\delta_{r_i}(\log r_i) = \log I_{DSi} - \log I_{DSi}(\log V_n, \log r_i) = 0 \qquad i = 1 \ldots M$$

B. Symmetry

All differential structures are designed symmetrically. This allows to reduce the degrees of freedom by introducing symmetry constraints. With a simulation-based DC formulation these symmetry constraints make the geometry of symmetric devices equal. The equivalent for an operating point driven DC formulation is making the node voltages and branch currents of symmetric nodes respectively symmetric branches equal.

VI. Example

As example, the CMOS current buffer OTA depicted in Fig. 4 [8] is used. The technology is 0.7μm Mietec CMOS. The list of specifications is given in table II. All transistors are biased in the saturation region. The power supply voltage is ± 2.5V. The load capacitance is 5pF. The independent input set is $\log\{ V_{IN\,DC}, V_{GS1a}, V_{GS2a}, V_{GS4a}, V_{GS5a}, V_{GS7}, V_{DS3a}, V_{DS3b}, V_{DS4a}, V_{OUT}, I_{DS1a}, I_{DS4a}, L_{M1a}, L_{M3a}, L_{M4a}, f_u, f_{gmr} \}$. Minimal area devices are M_{2a}, M_{5a}, M_6, and M_7. Symmetry constraints are $I_{DS1a} = I_{DS1b}$, $I_{DS4a} = I_{DS4b}$, $L_{M1a} = L_{M1b}$, $L_{M2a} = L_{M2b}$, $L_{M3a} = L_{M3b}$, $L_{M4a} = L_{M4b}$, $L_{M5a} = L_{M5b}$, $V_{n4a} = V_{n4b}$, $V_{in+} = V_{in-}$, $V_{n3} = V_{n\,out}$. This input set enforces the operating point, making it design oriented. Generating an initial starting point is straightforward. The low optimization times allow interactive use, which is not the case for batch oriented methods [3]. DC consistency is guaranteed by construction. With minimax optimization [1], a maximal unity gain frequency of 274MHz is obtained after around 100 evaluations. The performance parameters fulfill the specifications as depicted in table II.

VII. Conclusions

An efficient and complete DC modeling method has been presented. It is applicable to analog integrated CMOS circuits. The operating point is specified directly and the device sizes W and L are determined out of it. The algorithm lacks convergence problems, thereby guaranteeing algorithms like optimization a DC consistent result in all circumstances. It is computationally efficient with respect to traditional methods, making it suitable for interactive design space exploration using optimization-based sizing. It is suitable for parallellisation, allowing additional speedup.

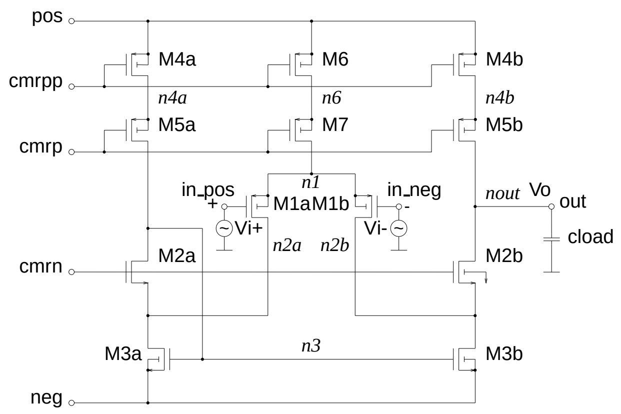

Fig. 4. Schematic of a current buffer OTA.

	required	obtained
I_{tot}	$\leq$ 3mA	3mA
PM	$\geq 60^\circ$	60°
GMR	$\geq$ 6dB	18.6dB
A_{V0}	$\geq$ 60dB	86.4dB
V_{off}	$\leq$ 5mV	5mV
SR	$\geq 150V/\mu s$	$150V/\mu s$
OR	$\geq \pm 1.5$V	1.5V
f_u	max	274MHz

TABLE II

Specifications and obtained performance of the current buffer OTA.

Acknowledgment

This work has been funded by the Flemish IWT under the MEDEA SADE program.

References

[1] F. Leyn, W. Daems, G. Gielen, and W. Sansen. A Behavioral Signal Path Modeling Methodology for Qualitative Insight in and Efficient Sizing of CMOS Opamps. In *Proceedings IEEE/ACM International Conference on Computer Aided Design*, pages 374–381, San Jose, CA, November 9–13, 1997. IEEE/ACM. ISBN 0-8186-8200-0.

[2] F. Leyn, W. Sansen, and G. Gielen. Transforming Small-Signal modeling into Control System Modeling. In *Proceedings Custom Integrated Circuits Conference*, pages 23.1.1–4, Santa Clara, CA, May 11–14, 1998. IEEE. ISBN 0-7803-4292-5.

[3] E. Ochotta, L. R. Carley, and R. Ruthenbar. ASTRX/OBLX: tools for rapid synthesis of high-performance analog circuits. In *Proceedings ACM/IEEE DAC*, pages 24–30. IEEE, 1994.

[4] J. Vlach and K. Singhal. *Computer methods for circuit analysis and design.* John Wiley & Sons Ltd., New York, 1974. ISBN 0-470-20850-3.

[5] P. C. Maulik and L. R. Carley. Automating analog circuit design using constrained optimization techniques. In *Proceedings IEEE/ACM International Conference on Computer Aided Design*. IEEE/ACM, November 1993.

[6] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C. The Art of Scientific Computing.* Cambridge University Press, Cambridge, 1991. ISBN 0-521-35465-X.

[7] B. Nichols, D. Buttlar, and J. P. Farrell. *Pthreads Programming.* O'Reilly&Associates, Inc., Sebastopol, CA,, 1996. ISBN 1-56592-115-1.

[8] John A. Fisher and Rudolf Koch. A highly linear CMOS buffer amplifier. *IEEE Journal of Solid-State Circuits*, 22(3):330–334, June 1987.