Automating the Sizing of Analog CMOS Circuits by Consideration of Structural Constraints

R. Schwencker[1,2], J. Eckmueller[2], H. Graeb[1], K. Antreich[1]

[1]Institute of Electronic Design Automation
Technical University of Munich
80290 Munich, Germany

[2]Siemens, Semiconductor Group
P.O. Box 80 17 09
81617 Munich, Germany

Abstract

In this paper, a method for the automatic sizing of analog integrated circuits is presented. Basic sizing rules, representing circuit knowledge, are set up before the sizing and are introduced as structural constraints into the sizing process. Systematic consideration of these structural constraints during the automatic sizing prevents pathologically sized circuits and speeds up the automatic sizing. The sizing is done with a sensitivity-based, iterative trust region method.

1 Introduction

Although the analog part of a mixed signal circuit is rather small, its design time often exceeds the time needed for the design of the digital part. The reason is, that for the digital part many design automation tools are available, whereas for the analog design numerical circuit simulators are still the most important tools. Circuit sizing is usually done manually by experienced analog designers and is therefore a time-consuming and error-prone task.

In the last years, symbolic[3] and numerical, simulation-based[5, 7] sizing tools were developed and are commercially available, but they are still rarely used.

For the simulation-based tools, the time consuming tasks are the circuit simulations, whereas the computational effort for the optimization algorithm itself can be neglected. Expensive optimization algorithms can be used, if the number of simulations can be reduced.

Moreover, the circuit designer usually specifies only those performances, that are critical for the application of the circuit. Thus, circuit specifications are often incomplete. Solving this incompletely defined problem with an automatic sizing tool often results in bad or pathologically sized circuits, that violate basic design rules. These circuits are very sensitive to variations of process and operational parameters and the unspecified performances may have unexpected values.

In this contribution, an automatic sizing method is presented, that overcomes these problems by introducing circuit knowledge into the sizing process. Basic sizing rules are set up on component level for transistor pairs (e.g. differential pair) and subcircuits (e.g. cascode current mirror) and formulated as constraints. These structural constraints express general function and matching conditions. By systematic consideration of structural constraints, pathologically sized circuits are prevented and so automatic sizing of analog circuits is enabled.

The sizing is done with a sensitivity-based, iterative trust region algorithm. In each step, the circuit performances are linearized and a parameter correction is calculated based on the characteristic boundary curve[1, 8]. Thus, the number of simulations can be reduced significantly.

2 Structural Constraints

Performances describe the circuit behavior. Specifications define requirements on the performances and thus on the circuit behavior. Normally, the circuit specifications and thus the requirements on the circuit behavior are incomplete. In an automatic sizing process this can result in a pathologically sized circuit that fulfills all given specifications, but violates basic design rules.

These basic design rules depend only on the structure of the circuit[2]. Therefore they are called structural constraints. As shown in Figure 1, these constraints express general function and matching conditions. By fulfilling the constraints, proper function of the circuit is guaranteed.

Structural constraints are derived on component level, for transistor pairs and subcircuits. At this level there is only a limited number of structures. These structures have a low complexity and are not specific for the circuit class. Thus, constraints can be derived for all relevant structures and therewith for every circuit.

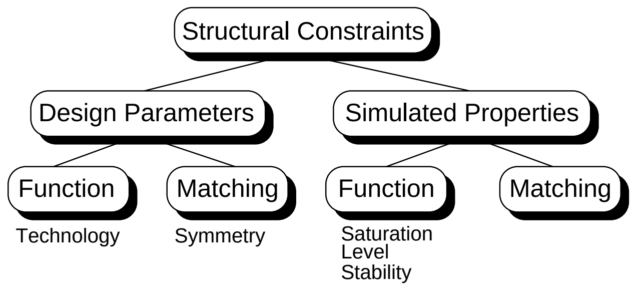

Figure 1. Structural constraints

Structural constraints for transistor pairs can be set up for differential pairs, level shifter, complementary pairs and current mirrors[2]. In this paragraph setting up structural constraints is shown on the example of a current mirror (Figure 2). The most important requirements, that a proper designed current mirror must fulfill are:

- For good mismatch properties and an area efficient layout, the length of the two transistors must be the same. With the transistor length being the same, the ratio of the two transistor widths must be equal to the ratio of the currents:

$$l_{\mathrm{M1}} = l_{\mathrm{M2}} \tag{1}$$
$$I_1/I_2 = w_{\mathrm{M1}}/w_{\mathrm{M2}} \tag{2}$$

- The smaller the area of a transistor, the higher is its mismatch sensitivity. Therefore, the transistor width and length must not fall below a minimum value w_{min} and l_{min}:

$$w_i \geq w_{min}, \quad \text{and} \tag{3}$$
$$l_i \geq l_{min}, \quad i \in \{\mathrm{M1,M2}\} \tag{4}$$

- Both transistors operate as voltage-controlled current sources (vccs) and thus they must be in saturation:

$$0 < V_{DS_i} < V_{G_i} = V_{GS_i} - V_T, \quad i \in \{\mathrm{M1,M2}\} \tag{5}$$

- For a low V_{T}-mismatch sensitivity, the effective gate voltage must not fall below a minimum value $V_{G,min}$:

$$0 < V_{G,min} < V_{GS,i} - V_{\mathrm{T}}, \quad i \in \{\mathrm{M1,M2}\} \tag{6}$$

- For a low λ sensitivity the difference of the drain-source voltages must not exceed a maximum value $V_{DS,max}$:

$$|V_{DS_{\mathrm{M1}}} - V_{DS_{\mathrm{M2}}}| < V_{DS,max} \tag{7}$$

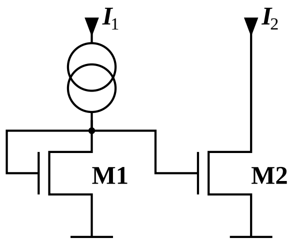

Figure 2. Current mirror

As shown above, structural constraints are either equality or inequality constraints. Equality constraints can only occur for design parameters and are easily fulfilled via tracking parameters. Therefore they are not included in the sizing algorithm. For each equality constraint the number of free design parameters is reduced by one.

Inequality constraints can occur for design parameters and simulated properties. They can be written as a function u_i, depending on the design parameters $\mathbf{d}$. For an easy notation they are formulated in a unique way:

$$u_i(\mathbf{d}) \geq 0, \quad \forall\ i \in [1, \ldots, n_u] \tag{8}$$

Combining all inequality constraints in the vector $\mathbf{u}$, (8) can be written as:

$$\mathbf{u}(\mathbf{d}) \geq \mathbf{0} \tag{9}$$

This inequation defines the feasibility region of the sizing algorithm discussed in the next section.

3 Automatic Sizing

For each performance f_i, a specification $f_{s,i}$ is given by the designer. The design parameters $\mathbf{d}$ should be tuned, so that the sized circuit satisfies the given specification[1]:

$$f_i(\mathbf{d}) \overset{!}{=} f_{s,i} \quad \forall\ i \in [i, \ldots, n_f] \tag{10}$$

For an easy notation, all performances and specifications are combined in the vectors $\mathbf{f}$ and $\mathbf{f}_s$. The difference between the performances and specifications is the error $\boldsymbol{\epsilon}$ of the circuit:

$$\boldsymbol{\epsilon}(\mathbf{d}) = \mathbf{f}(\mathbf{d}) - \mathbf{f}_s \tag{11}$$

Beside the specifications, the structural constraints (9) must also be fulfilled. The resulting problem is a constrained nonlinear minimization problem:

$$\min_{\mathbf{d}} \left\{ ||\boldsymbol{\epsilon}(\mathbf{d})||^2 \mid \mathbf{u}(\mathbf{d}) \geq \mathbf{0} \right\} \tag{12}$$

[1]The specification can be tightened or relaxed during the optimization in order to adapt the optimization process to the performance potential turning out during the optimization.

As circuit simulations are extremely expensive, an optimization algorithm is required, that needs few simulations. The algorithm presented in the rest of this section carefully plans each iteration step, investigating optimization cost (in form of the required correction norm) and optimization effectiveness (in form of the achievable error reduction towards the specification).

3.1 Sizing Algorithm

The minimization problem is solved with an iterative algorithm. In every step μ, the performances are linearized at the current nominal point $\mathbf{d}_0^{(\mu)}$:

$$\begin{aligned}\mathbf{f}(\mathbf{d}) &= \underbrace{\mathbf{f}(\mathbf{d}_0^{(\mu)})}_{\mathbf{f}_0^{(\mu)}} + \underbrace{\nabla_{\mathbf{d}}\mathbf{f}|_{\mathbf{d}=\mathbf{d}_0^{(\mu)}}}_{\mathbf{S}^{(\mu)}} \cdot \underbrace{(\mathbf{d}-\mathbf{d}_0^{(\mu)})}_{\mathbf{x}} + \ldots \\ \hat{\mathbf{f}}(\mathbf{x}) &= \mathbf{f}_0^{(\mu)} + \mathbf{S}^{(\mu)} \cdot \mathbf{x}\end{aligned} \tag{13}$$

The same is done with the constraints:

$$\begin{aligned}\mathbf{u}(\mathbf{d}) &= \underbrace{\mathbf{u}(\mathbf{d}_0^{(\mu)})}_{\mathbf{u}_0^{(\mu)}} + \underbrace{\nabla_{\mathbf{d}}\mathbf{u}|_{\mathbf{d}=\mathbf{d}_0^{(\mu)}}}_{\mathbf{U}^{(\mu)}} \cdot \underbrace{(\mathbf{d}-\mathbf{d}_0^{(\mu)})}_{\mathbf{x}} + \ldots \\ \hat{\mathbf{u}}(\mathbf{x}) &= \mathbf{u}_0^{(\mu)} + \mathbf{U}^{(\mu)} \cdot \mathbf{x}\end{aligned} \tag{14}$$

The linearized error is defined according to (11):

$$\hat{\boldsymbol{\epsilon}}(\mathbf{x}) = \mathbf{S}^{(\mu)}\mathbf{x} + (\mathbf{f}_0^{(\mu)} - \mathbf{f}_s) = \mathbf{S}^{(\mu)}\mathbf{x} + \boldsymbol{\epsilon}_0^{(\mu)} \tag{15}$$

The sizing is done with the following algorithm based on the linearization discussed above:

1. Set the iteration index $\mu = 0$ and the nominal point $\mathbf{d}_0^{(\mu)}$ to the given initial parameters.

2. Calculate the sensitivity matrixes $\mathbf{S}^{(\mu)}$ and $\mathbf{U}^{(\mu)}$, the performance values $\mathbf{f}_0^{(\mu)}$ and the constraint values $\mathbf{u}_0^{(\mu)}$ as described in (13) and (14).

3. Calculate a parameter correction $\mathbf{x}$ as shown in section 3.2 and a new nominal point:
$$\mathbf{d}_0^{(\mu+1)} = \mathbf{d}_0^{(\mu)} + \mathbf{x} \tag{16}$$

4. Check, if the linearization is still valid. This is done by comparing the error reduction in the linear and the nonlinear system:
$$\rho = \frac{||\boldsymbol{\epsilon}_0^{(\mu)}|| - ||\boldsymbol{\epsilon}(\mathbf{d}_0^{(\mu+1)})||}{||\boldsymbol{\epsilon}_0^{(\mu)}|| - ||\hat{\boldsymbol{\epsilon}}(\mathbf{x})||} \geq \rho_0, \quad \rho_0 \in]1,0[\tag{17}$$
Hereby ρ_0 is a properly chosen constant. If (17) holds, the new nominal point $\mathbf{d}_0^{(\mu+1)}$ is accepted, otherwise it is rejected. In the latter case the linearization is not valid. A new parameter correction with a smaller $||\mathbf{x}||$ is calculated and the algorithm continues with step 3.

5. The algorithm terminates with the solution $\mathbf{d}_0^{(\mu+1)}$ if either
$$\begin{aligned}||\boldsymbol{\epsilon}_0^{(\mu)} - \boldsymbol{\epsilon}(\mathbf{d}_0^{(\mu+1)})|| &\leq \tau_0 \quad \text{or} && (18)\\ ||\boldsymbol{\epsilon}(\mathbf{d}_0^{(\mu+1)})|| &\leq \tau_0, \quad \tau_0 > 0 && (19)\end{aligned}$$
is fulfilled. τ_0 is the maximum acceptable error. If (18) holds, no more improvement is possible – a local or global minimum is reached. (19) means, that the specification is fulfilled.

6. Increase the iteration index μ and continue with step 2.

For this algorithm, convergence to a local minimum of $||\boldsymbol{\epsilon}(\mathbf{d})||$ can be proven theoretically[4]. Circuit examples presented in section 4 also show the effectiveness of this method in practice.

3.2 One parameter correction step

In this section, it is shown, how the parameter correction $\mathbf{x}$, needed in step 3 of the sizing process is calculated. All calculations are done in the linearized system in one iteration of the sizing algorithm. Thus, the iteration index μ will be left out in this section.

In the linearized system (12) can be written as:

$$\min_{\mathbf{x}} \left\{ ||\boldsymbol{\epsilon}_0 + \mathbf{S}\mathbf{x}||^2 \mid \mathbf{u}_0 + \mathbf{U}\mathbf{x} \geq \mathbf{0} \right\} \tag{20}$$

This problem can be solved exactly, but if the matrix $\mathbf{S}$ is ill conditioned, huge parameter corrections $||\mathbf{x}||$ may occur. In this case it can happen, that the linearization is not valid, and so, although $\mathbf{x}$ solves (20), the error in the nonlinear system (11) may even raise or the simulator does not converge with the new parameters.

To avoid this problem, an objective function is defined, that takes error reduction and parameter correction into account[6, 8]:

$$\Phi(\mathbf{x}, \lambda) = ||\boldsymbol{\epsilon}_0 + \mathbf{S}\mathbf{x}||^2 + \lambda||\mathbf{x}||^2, \quad \lambda \geq 0 \tag{21}$$

With this objective function the new minimization problem can be set up. Solving this problem we get the solution $\mathbf{x}_c(\lambda)$ and its error $\boldsymbol{\epsilon}_c(\lambda)$ in the linear system:

$$\begin{aligned}\mathbf{x}_c(\lambda) &= \operatorname*{argmin}_{\mathbf{x}} \{\Phi(\mathbf{x}, \lambda) \mid \mathbf{u}_0 + \mathbf{U}\mathbf{x} \geq \mathbf{0}\} && (22)\\ \boldsymbol{\epsilon}_c(\lambda) &= \boldsymbol{\epsilon}_0 + \mathbf{S}\mathbf{x}_c(\lambda) && (23)\end{aligned}$$

The index "c" denotes, that this is the solution of a constrained problem. The solution can be transformed as discussed in[8]. The transformed correction norm a_c is:

$$a_c(\lambda) = \frac{||\mathbf{x}_c(\lambda)|| - ||\mathbf{x}_c(\lambda \to \infty)||}{||\mathbf{x}_c(0)|| - ||\mathbf{x}_c(\lambda \to \infty)||} \tag{24}$$

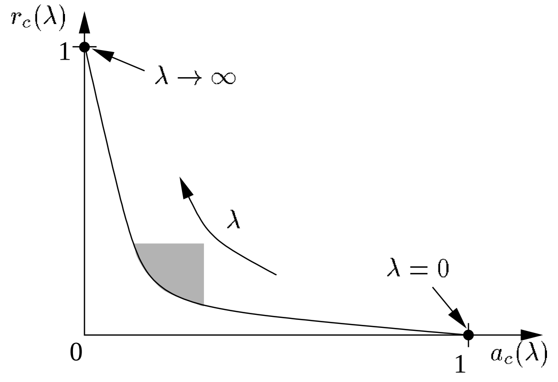

Figure 3. Typical example of a characteristic boundary curve (CBC).

The transformed error norm r_c is calculated as

$$r_c(\lambda) = \frac{\breve{r}(\lambda) - r_{min0}}{1 - r_{min0}} \tag{25}$$

with

$$\breve{r}(\lambda) = \sqrt{\frac{||\epsilon_c(\lambda)||^2 - ||\epsilon_r||^2}{||\epsilon_c(\infty))||^2 - ||\epsilon_r||^2}}, \quad \text{and} \tag{26}$$

$$r_{min0} = \breve{r}(0) \tag{27}$$

Hereby $||\epsilon_r||$ is the residual of the unconstrained minimization problem

$$||\epsilon_r|| = \min_{\mathbf{x}} \{||\epsilon_0 + \mathbf{Sx}||\} . \tag{28}$$

The resulting curve $r_c(a_c)$ is the characteristic boundary curve (CBC)[8, 1]. Figure 3 shows a typical CBC. The curve has the following properties:

- The transformed error r_c and correction norm a_c are in the intervall $[0, 1]$. Thus, different CBCs are comparable.
- The CBC is the solution with the smallest error $||\hat{\epsilon}(\mathbf{x})||$ for a given maximum correction norm. This means, that no solutions below the CBC are possible.
- For a sensitivity matrix $\mathbf{S}$ with an ideal condition number of one and no active constraints, the curve is a straight line. The higher the condition number of the matrix, the more bend is the curve. In every case the curve is convex.

Efficient ways for calculating the characteristic boundary curve are presented in[8, 1].

It can be seen in Figure 3, that usually with a small correction a_c, the error r_c can be reduced significantly, whereas a huge correction is needed to reduce the error to zero. The goal is, to find a solution with a good ratio between correction effort and correction effect. These solutions are near the bend of the curve, marked with a grey color in the figure. The bend can be found based on the curvature-radius κ of the curve:

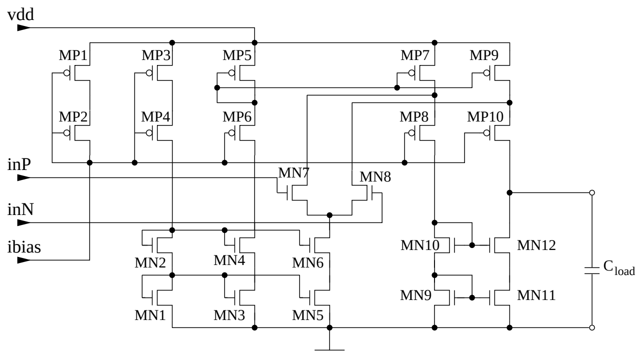

Figure 4. Folded-cascode operational amplifier.

$$\kappa = \frac{\sqrt{\left(1 + \left(\frac{\partial r_c}{\partial a_c}\right)^2\right)^3}}{\frac{\partial^2 r_c}{\partial a_c^2}} \tag{29}$$

The smaller the radius κ, the higher the curvature. So the solution with the smallest curvature-radius is taken as the parameter correction $\mathbf{x}$ for this step. The curvature and curvature-radius of the CBC can efficiently be calculated by interpolating the CBC between two calculated points with a Berstein-Bézier-curve (B-B-curve). The B-B-curve is a parametric curve, that, if appropriately calculated, has the following properties:

- It is convex,
- in the starting and ending point it has the same slope as the CBC and
- all derivatives can be calculated analytically.

The first two properties guarantee, that the B-B-curve principally has the same shape as the CBC and converges to the CBC for small differences between the starting and ending point. The last property is important for an efficient calculation of the curvature radius.

With this algorithm, a parameter correction with a good ratio between correction effort and correction effect can be calculated with an acceptable computational effort.

4 Results

The introduced automatic sizing method was used to size the operational amplifier shown in Figure 4. The circuit consists of 22 MOS-transistors. With two design parameters (width and length) for every transistor this results in 44 design parameters.

For this circuit 165 inequality and 35 equality constraints can be derived. As stated in section 2, every equality constraint reduces the number of free design parameters by one, which is resulting in 9 free design parameters or a reduction of 80%.

All structural constraints are evaluted with an operating point simulation, that must be done in any case. Thus, no additional simulation effort is needed for the evaluation of the constraints.

Starting from a pathologically sized initial point, the circuit was sized twice. Once in consideration of all structural constraints and once only with respect to parameter bounds. Table 1 compares the given specifications and the achieved results.

	Specification	Initial sizing	With constr.	Without constr.
A_0 [dB]	75	99.5	74.7	75.5
f_t [MHz]	54	7.9	54.1	53.9
PHM [°]	70	34.0	69.9	69.9
SR_p [V/μs]	42	4.4	41.9	42.6
Power [mW]	1.7	0.36	1.71	1.71
Violated constr.	—	6	0	7
Iterations	—	—	5	14
Sensitivity calc.	—	—	45	126

Table 1. Results for the operational amplifier.

The sensitivities were calculated with finite differences, using the forward-difference formula. For this calculation method a simulation time of approximately 240s on a SUN Ultra 1 was needed for each iteration. This is resulting in a sizing time of 20min for the constrained circuit and nearly one hour in the unconstrained case. In any case the CPU-time needed by the optimization algorithm itself is neglegible. The total sizing time could further be reduced by using a simulator with built-in sensitivities.

It clearly turns out, that considering the constraints significantly speeds up the sizing algorithm. Furthermore, the circuit sized without consideration of the structural constraints violates several basic design rules. Especially the effective gate voltages of the transistors MN5, MN1 and MN3 are too small. These low effective gate voltages result in a mismatch-sensitive circuit as shown on the example of the CMRR in table 2.

Performance	With Constr.		Without Constr.	
	mean	σ	mean	σ
CMRR [dB]	98.6	9.6	92.2	9.7

Table 2. Mean values and standard deviations σ calculated with a 100 sample monte carlo analysis with V_T variations.

In summary it emerges, that, although both sizings fulfill the specification, the circuit sized under consideration of the structural constraints is significantly less sensitive to process variations.

5 Conclusion

In this paper, a method for the automatic sizing of integrated analog CMOS circuits is presented. Basic sizing rules for transistor pairs (e.g. differential pair) and subcircuits (e.g. cascode current mirror) are set up on component level and formulated as constraints. A systematic consideration of these structural constraints during the sizing significantly reduces the number of free design parameters, speeds up the sizing, and prevents pathologically sized circuits. Setting up the structural constraints is shown on the example of a current mirror.

The sizing is done with an iterative trust region algorithm. In each step, the circuit performances and constraints are linearized and a parameter correction with a good ratio between error reduction and parameter deviation is calculated based on the characteristic boundary curve.

The sizing method was applied to a folded-cascode operational amplifier. It turned out, that by systematic consideration of the structural constraints, the overall sizing time was reduced by a factor of three and the resulting circuit is less sensitive to process variations.

Acknowledgement

We would like to thank Mr. A. Sanders and Dr. R. Koch of Siemens Semiconductor for promoting this work.

References

[1] K. J. Antreich, P. Leibner, and F. Pörnbacher. Nominal design of integrated circuits on circuit level by interactive optimization. *IEEE Trans. Circuits and Systems (CAS)*, 35:1501–1511, 1988.

[2] J. Eckmüller, M. Gröpl, and H. Graeb. Hierarchical characterization of analog integrated CMOS circuits. In *DATE*, 1998.

[3] G. Gielen, H. Walscharts, and W. Sansen. Analog circuit design optimization based on symbolic simulation and simulated annealing. *IEEE Journal on Solid-State Circuits*, 25:707–713, 1990.

[4] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press. Inc., London, 1981.

[5] A. E. E. S. GmbH. *OPSIM User's Manual*. Ulm, 1993.

[6] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Ind. App. Math.*, 11:431–441, 1963.

[7] W. Nye, D. Riley, A. Sangiovanni-Vincentelli, and A. Tits. Delight.spice: An optimization-based system for the design of integrated circuits. *IEEE Trans. on Computer-Aided Design (CAD)*, 7:501–519, 1988.

[8] F. Pörnbacher. A new method supporting the nominal design of analog integrated circuits with regard to constraints. In *ECCTD*, pages 614–618, 1989.