

Analog IP Porting by Topology Conversion and Optimization

Udo Sobe¹, Achim Graupner¹, Enno Böhme¹, Andreas Ripp² and Michael Pronath²

¹ZMD AG
Grenzstraße 28
01109 Dresden
{Udo.Sobe, Achim.Graupner, Enno.Boehme}@zmdi.com

²MunEDA GmbH
Stefan-George-Ring 29
81929 München
{Andreas.Ripp, Michael.Pronath}@muneda.com

Abstract

Design reuse is usually performed on a small scale in analog design. Proven design topologies and concepts are recycled by duplication, modification and development during conversion. A lot of repeated manual and interactive tasks dominate the process to transfer the design data between technologies. Hence tool support to reduce failures during conversion is necessary.

Further effort is required for device sizing to ensure the circuit meets the previous or a changed specification. There, optimization tools can support the designer.

We present an approach to convert a circuit design from technology A to technology B. It is separated into topology conversion to transfer design data and an optimization step for sizing.

1 Introduction

The strategy of fabless design gives the freedom of choice of a suitable technology for the IC design. This flexibility is not built into the analog circuit design, because they are developed and optimized for one specific technology A (source technology). Frequently the sizing of the devices prevents the direct reuse of analog circuit for another technology B (destination technology).

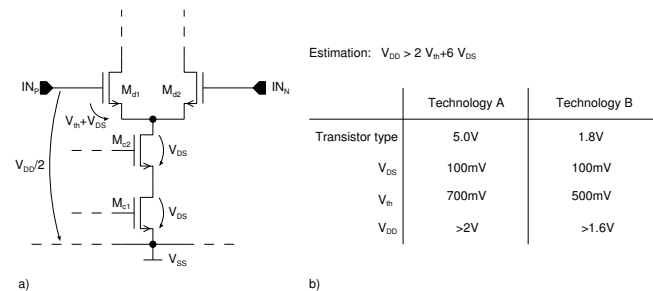
This is why many reuse approaches consider the circuit topology as the real analog IP (intellectual property). So an interesting solution, which is still under development, is to generate technology independent topologies and provide such IP in a library [1]. A further step towards flexibility can be taken by decoupling topology data from the design environment [2].

In practice analog designers tend to reuse a known circuit topology of a proven design. This starting point allows to check the performance of the circuit topology using a different technology B or to develop the circuit topology further. Additional modifications are necessary [3] due to change of the specification regarding the performance parameters or implementation of new features.

A lot of design data of the circuit topology are bound to the topology's PDK (process development kit) inside the design environment such as Cadence [4]. Hence the transfer of design data from technology A to technology B is associated with design data transfer from PDK A to a different PDK B. The design environment we use, the Cadence Design Framework, does not provide an efficient function for this; the search and replace function lacks the required functionality.

We know that the reuse of a given circuit topology is limited by many conditions. Hence, a feasibility estimation (cf. Fig. 1) should be the first step. This can be done using simplified models of the topology. One challenge of conversion is the reduced range of operating condition of the devices, e.g. the transition from 5V devices to 3.3V devices.

This article describes a method to convert and optimize a circuit topology. Because the method reflects common design practice, it is important to improve tool support. Therefore additional details to transfer design data inside the Cadence design environment are given. Then the optimization strategy is presented using MunEDA's optimization tool suite WiCkeD [5].



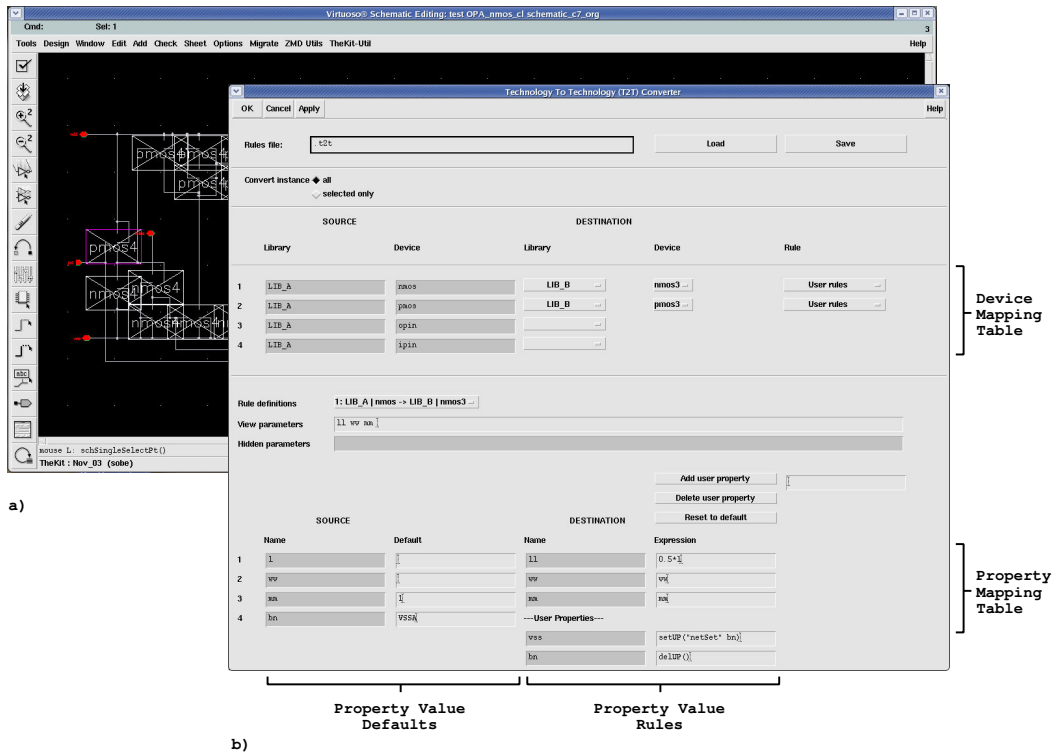


Figure 2: Screenshot of the T2T converter window and schematic window. a) Instantiated devices are display as dummies due to different library and/or cell names in the PDK B. b) T2T converter GUI with comments.

user property is supported (see example in section 5). Design variables and pPar parameters can be transferred too. T2T does not adapt symbol differences between SRC and DST. The transfer of performance parameters (e.g. gain, unity gain bandwidth) is not objective of the T2T tool (see section 3).

Once the T2T setup is done, it can be used to convert complete libraries. With an appropriate setup, this can also be used to convert designs to a PDK independent device library [1].

The conversion can reduce the circuit performance or even cause the circuit not to work at all, because the circuit performance is not considered for sizing at this stage. Therefore

the functionality has to be analyzed.

Generally, the circuit must be resized to restore its performance or to meet a new specification. The traditional approach is manual sizing by the designer. MunEDA's optimization suite WiCkeD [5] is convenient for this task, because it supports an interactive optimization approach (see section 4).

A final verification step is used to check performance over PVT conditions (Process, operating conditions: Voltage and Temperature) with corner and Monte Carlo analyses before the layout is created. Our in-house verification environment called zmdAnalyser [6] is used for this step.

The flow is also suitable to transfer design data between different PDK versions of one technology. In this case the optimization step may be skipped.

3 Topology Conversion

Fig. 4 shows the conversion steps. First an instance list is built from circuit's schematic view. It is possible to convert all instances or only those selected in the Virtuoso Schematic Editor window.

The devices are converted to the new PDK B according to the device mapping table configured in the GUI. This step is simply a search and replace of two parameters: library name and cell name.

Parameter names may differ from PDK to PDK. So a parameter name mapping is necessary for conversion, e.g. *w* in PDK A to *width* in PDK B.

Besides the parameter name the parameter value has to be set. Different rules supporting various scenarios are supported, such as:

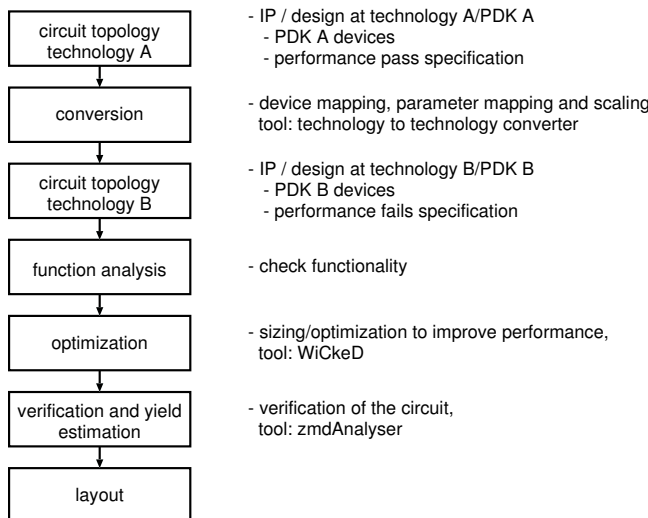


Figure 3: Out porting strategy is focused on daily design tasks.

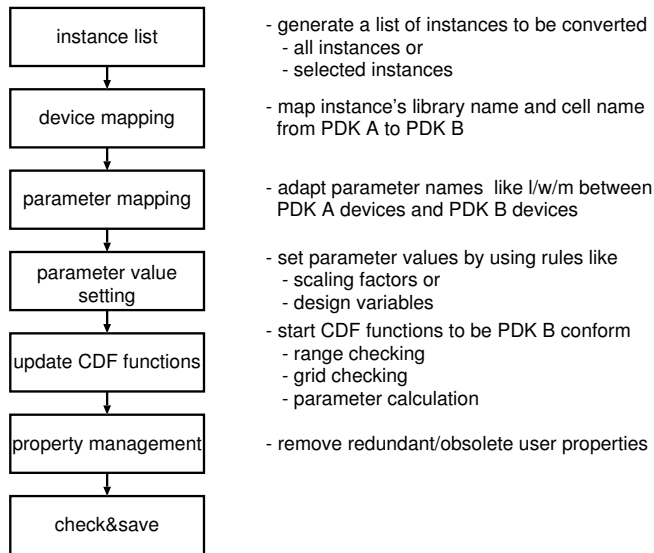


Figure 4: Details of transfer design data to another PDK.

- use a parameter of PDK A, e.g. $l := l$
- scale a parameter of PDK A by a factor, e.g. $l := 0.5 \cdot l$
- calculation by using different parameters of PDK A, e.g. $m := m \cdot fingers$
- set a fixed value, e.g. $l := 0.35u$
- set a design variable, e.g. $l := "myLength"$
- use a decision tree, e.g. $l := \text{if}(l == 0.6u \text{ then } 0.35u \text{ else } l)$

An important detail for the design data transfer is the default setup of the Cadence design environment for parameter storage: Property values which are set to the PDK's defaults are not stored as instance parameters. The GUI does not determine the defaults from the PDK but provides fields for default parameter values, which are used when the property is not defined.

The parameter mapping and scaling is validated by updating the CDF (component description format) functions. These functions are provided by today's PDKs and realize a lot of PDK-related tasks in the background.

CDF parameters are used to create and describe every component (e.g. nmos, pmos, amplifier) in the Cadence design environment. A CDF callback can be defined for each CDF parameter using SKILL expressions. CDF callbacks are carried out during parameter entry in the object property form. CDF callbacks are used in current PDKs to

- validate entered values (e.g. check limits such as minimal length)
- control form appearance (e.g. calculate method for resistors)
- compute the values of other CDF parameters for
 - simulation
 - physical verification
 - pcell (i.e. calculate resistor length from resistance and width)

The procedure used to trigger CDF callback functions is technology and PDK independent. CDF callbacks are triggered in the sequence of their definition and update the parameters mentioned above.

An additional property management step is necessary to control user properties. In our experience many schematics contain obsolete data, which was accumulated over the cell's history and should be deleted at this point.

The conversion is finished by the final check and save step. In this step, standard checks grouped into logical, physical, name, inherited connection and AMS checks are performed.

4 Sizing by Optimization

Our approach to convert the circuit topology prepares the design data for PDK B in a GUI-based design environment, which dominates our reuse practice. Unlike other solutions as [7], this first step only transfers design data with an optional scaling of parameters without any focus on the circuit's performance. The latter is achieved by optimization during the sizing process using an interactive optimization approach.

The optimization consists of three main steps (cf. Fig. 5). First the operating points of basic structures, e.g. differential pairs, are optimized using a constraint matrix concept. Then the circuit performance is improved regarding operating range, such as supply voltage and temperature. Finally design centering is carried out to maximize the yield.

4.1 Structural Constraints

Analog circuits are composed of basic building blocks like current mirrors or differential pairs. Unlike digital gates, the analog ones depend on their geometries and operating point to operate correctly. Usually, being in saturation is an important constraint on many analog transistors, as well as current symmetries or certain nodes being at the same potential. Since these constraints are neither related to the specification, nor to the layout level like design rules are, but come from the structure of the circuit, they are called "structural constraints". We distinguish four types of constraints:

1. geometric equality, like "equal lengths $l_1 = l_2$ in a current mirror"
2. geometric inequality, like " $w_1 l_1 > 6L_{\min}^2$ "
3. electrical equality, like " $I_1 = I_2$ " in a current mirror
4. electrical inequality, like " $V_{gs} - V_{th} > 50mV$ " (strong inversion)

The geometric constraints can be guaranteed by construction. To check the electrical inequality constraints, a simulation has to be done that shows by which amount c_k each constraint k is over-fulfilled ($c_k > 0$) or is violated ($c_k < 0$).

Like design rules on layout level, structural constraints on schematic level do not at all guarantee that the circuit fulfills the specification, but a violation indicates a structural problem that may result in a low yield but remains undetected when simulating a rather high-level circuit specification.

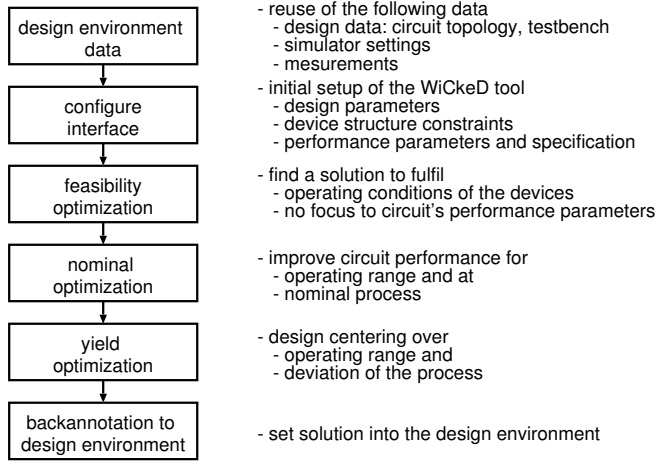


Figure 5: WiCkeD's main optimizations steps.

For a typical analog circuit consisting of 100 transistors, more than 400 constraints may be created. Since many structural constraints can be derived from requirements on basic structures like current mirrors, generation of many of these constraints can be performed automatically [8].

4.2 Feasibility Optimization (FO)

Structural constraints are useful to automatically find a good initial sizing and to ensure that tools for automatic nominal optimization and design centering provide technically feasible results [9]. For that purpose, FO modifies the vector $\mathbf{d} = (d_1, \dots, d_{n_d})$ of design parameters (like transistor geometries and resistor values) so that all constraints are fulfilled, i.e., $\mathbf{c}(\mathbf{d}) \geq \mathbf{0}$. Usually, a reasonable initial sizing \mathbf{d}_{init} is available and a solution close to it is preferred:

$$\begin{aligned} \min_{\mathbf{d}} \|\mathbf{d} - \mathbf{d}_{\text{init}}\| \\ \mathbf{c}(\mathbf{d}) \geq \mathbf{0}. \end{aligned} \quad (1)$$

The number of independent design parameters n_d grows with the number of elements to be sized and is reduced by geometric equality constraints. For typical analog circuits n_d can be expected to be between 15 and 30, while complex designs, e.g. the OTA presented in [10], can have up to 100 degrees of freedom.

4.3 Nominal Optimization (NO)

Analog circuits are characterized by performance measures, for example, gain A_0 , slew rate SR, noise figure NF. The specification requires the values of these measures not to exceed certain upper and/or lower bounds, for example $A_0 \geq 80$ dB.

We denote the performance measures by the vector $\mathbf{f} = (f_1, \dots, f_{n_f})$, with the vectors of lower bounds \mathbf{f}^L and upper bounds \mathbf{f}^U . The performance measures depend on design parameters: $\mathbf{f}(\mathbf{d})$, and the specification is

$$\mathbf{c}(\mathbf{d}) \geq \mathbf{0} \quad \wedge \quad \mathbf{f}^L \leq \mathbf{f}(\mathbf{d}) \leq \mathbf{f}^U. \quad (2)$$

The goal of nominal optimization is finding values for \mathbf{d} that satisfy (2).

Moreover, this must be achieved for a defined range of operating parameters like temperature or Vdd. We denote the operating parameters by the vector $\boldsymbol{\theta} = (\theta_1, \dots, \theta_{n_\theta})$ with lower and upper bounds $\boldsymbol{\theta}^L$ and $\boldsymbol{\theta}^U$. Then, \mathbf{f} depends also on the operating conditions: $\mathbf{f}(\mathbf{d}, \boldsymbol{\theta})$, and the specification is

$$\mathbf{c}(\mathbf{d}) \geq \mathbf{0} \quad \wedge \quad \forall_{\boldsymbol{\theta}^L \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}^U} \mathbf{f}^L \leq \mathbf{f}(\mathbf{d}, \boldsymbol{\theta}) \leq \mathbf{f}^U. \quad (3)$$

The goal of nominal optimization with operating conditions is finding values for \mathbf{d} that satisfy (3).

Two types of algorithms are available for nominal optimization in WiCkeD: gradient-based optimization with parameter distances [11] and stochastic (global) optimization. The nature of most analog sizing problems is optimization of performance functions that show strong trade-offs, are expensive to evaluate in terms of simulation time, but are monotonous or convex—not in the full design space but in the small feasible design space that is restricted by the large number of structural inequality constraints. Gradient-based methods can be adapted to this type of problem very efficiently. Therefore it is reasonable to run these methods first, and to resort to stochastic optimizers only when simulation results and design knowledge indicate that multiple local optima actually exist.

4.4 Design Centering

Process variation and mismatch have a large influence on the performance measures of analog circuits. For simulation, this effect is modeled by varying randomly a few standard Gaussian distributed model parameters, for example t_{ox} or V_{th} . The vector of random model parameters is denoted by $\mathbf{s} = (s_1, \dots, s_{n_s})$ with the null vector $\mathbf{0}$ as mean and unity covariance matrix. Process variation and mismatch are both contained in \mathbf{s} , so for a typical analog circuit consisting of 100 transistors, n_s can be expected to be between 200 and 250.

One standard method for estimating the distributions of performance measures is Monte Carlo simulation. A sample of size N of \mathbf{s} is generated and simulated, yielding N result vectors $\mathbf{f}^{(i)} = \mathbf{f}(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}^{(i)})$, $i = 1 \dots N$. The parametric yield Y is estimated as the percentage of samples that lie within the specification bounds ($\mathbf{f}^L, \mathbf{f}^U$). Monte Carlo is only an analysis method, but does not vary \mathbf{d} and hence shows little information on how to improve the yield by changing \mathbf{d} .

Yield improvement can be accomplished by worst-case distance methods. A design that satisfies (3), i.e., that fulfills the specification for the typical process and no mismatch ($\mathbf{s} = \mathbf{0}$) and for all required operating conditions, could still violate the specification for some $\mathbf{s} \neq \mathbf{0}$. If process conditions \mathbf{s} causing violations are close to the mean value (i.e., $f_i(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}) < f_i^L$ for some $\boldsymbol{\theta}$ and small $\|\mathbf{s}\|$), then there will be severe parametric yield loss. Therefore, an important measure for a performance f_i is the worst-case distance $\beta_{\text{wc}}^{(i)}$, which is the shortest distance between the mean value and a process condition that causes $f_i(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s})$ to fail its specifica-

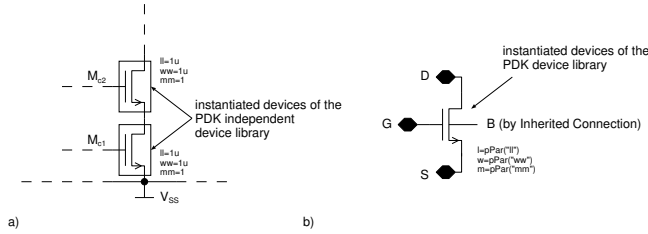


Figure 6: PDK independent device library. a) Application example in the circuit topology. b) Deposit PDK relevant data to a PDK independent device library.

tion. For a lower bound f_i^L of a spec that satisfies (3),

$$\beta_{wc}^{(i)} = \min_{\theta, s} |\beta|$$

$$f_i(\mathbf{d}, \theta, \beta \frac{\mathbf{s}}{\|\mathbf{s}\|}) = f_i^L \quad (4)$$

$$\theta^L \leq \theta \leq \theta^U.$$

The worst-case distance for an upper bound is similarly defined.

A worst-case distance is a function of the design parameters. They are useful goals to maximize over \mathbf{d} and thereby achieve a design that is centered in the process space regarding the specification bounds [12, 13].

5 PDK independent Device Library

A PDK independent topology library (e.g. [1]) decouples the device library and the technology PDK and simplifies IP porting. Only one topology conversion is necessary to prepare a topology for the PDK independent library. To reuse a topology in a new technology B, it must be converted to PDK B or into a PDK independent topology using a PDK independent device library mapped to the relevant PDK data.

One possible realization of the PDK independent device library and the application in the schematic is shown in Fig. 6. The devices are instantiated as usual (cf. cascode structure in Fig. 6a). An additional level of hierarchy instantiating PDK devices (see example in Fig. 6b) is used to perform the mapping to the PDK.

The example in Fig. 6 shows how symbols can be adapted. The circuit topology uses three-terminal devices, which were available in the PDK A device library, while PDK B provides four-terminal devices only. The used PDK independent device library contains three-terminal devices, which use a user property of type "netSet" at the instance to connect the bulk node using the inherited connection function of the Cadence design environment.

Some motivations to use a PDK independent device library are

- standard device names,
- common symbols (e.g. shapes and terminals),
- uniform parameter names and
- simplification of tool configuration and application due to the items listed above, e.g. configuration of WiCkeD.

Here the main effort to create a PDK independent device library is the generation of the additional hierarchy level.

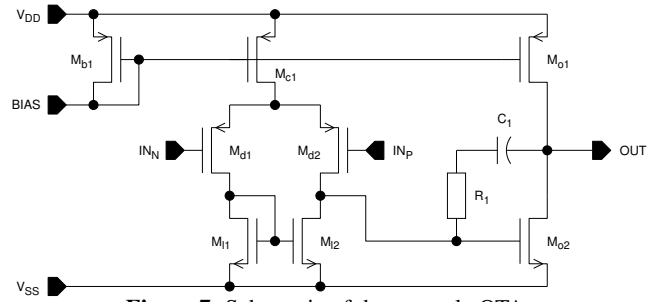


Figure 7: Schematic of the example OTA.

6 Example for the T2T Conversion

The T2T converter has already been used to transfer e.g. an analog front end to a new technology. However, most frequently it is used to generate just the required operational amplifiers.

We use the simple textbook operational transconductance amplifier (OTA) shown in Fig. 7 to illustrate a conversion from an $0.6 \mu\text{m}$ technology ($V_{DD} = 3.3\text{--}5 \text{ V}$) to an $0.18 \mu\text{m}$ technology ($V_{DD} = 1.8 \text{ V}$). In this example the focus will be to save area, reduce the power supply voltage and increase the bandwidth by a factor of 3 while keeping the remaining specification parameters except the gain (60 dB instead of 70 dB) the same.

As already mentioned, the T2T converter allows to limit conversion to the devices selected in the schematic. To illustrate this feature, the differential pair was converted to medium-V_{th} transistors while all other transistor were converted to standard devices. This was achieved by first converting the differential pair by selection using a setup converting to medium-V_{th} devices and then all other devices of the source PDK using the standard setup. All setups can be saved and reloaded to simplify reuse.

For the initial sizing we scaled all transistor lengths by a factor of 0.25 and set W/L to 1 using the property setup of the T2T converter; this is done without any regard for the circuit performance. Therefore, the resulting performance is very poor (see step 1 in Table 1).

Then we used WiCkeD to optimize the sizing. Since WiCkeD's optimization algorithm works better if the design parameters are independent, we used W/L and L instead of W and L as transistor design parameters.

For this circuit we obtained 53 structural constraints and 12 degrees of freedom (the ten independent design parameters of the transistors and one each for R and C). Then we employed Feasibility Optimization (FO), Nominal Optimization (NO) and Design Centering (also called Yield Optimization [YO]) to size the design.

The FO ensures that all constraints are met; as step 2 in Table 1 shows, this alone happens to improve most performances, although we still do not consider them in this step. The runtime of FO is less than 5 minutes. Note that we check constraints only at nominal operating conditions, while the performances must obviously be checked at their worst-case (wc) operating point.

After we perform the NO at nominal operating conditions, the phase margin does not yet meet the specification at worst-case operating conditions. To remedy this, we can run an

Design Step	Gain	BW	PM	I/I_{max}	SR	FC	Area
initial design (0.6 μm)	76.2	1.27	66.5°	0.978	1.632	0	100%
1 after T2T (0.18 μm)	20.6	0.24	26.1°	0.784	1.304	4	
2 after FO	54.7	1.45	27.1°	0.683	0.981	0	
3 after typ. NO	66.0	3.19	52.2°	0.993	1.289	0	
4 after w_c NO	69.8	3.11	62.1°	0.902	1.273	0	
5 after YO	68.4	3.25	67.3°	0.938	1.338	0	3.2%

Table 1: Important performance parameters at different stages of technology conversion. For each performance, the value at its respective worst-case operating condition is given.

Note: Gain = Open-loop Gain in dB; BW = Unity Gain Bandwidth, normalized; PM = Phase Margin; I/I_{max} = Current, normalized; SR = Slew Rate, normalized; FC = number of violated constraints; Area = relative Transistor Area (without area for R and C). Normalizations are all with respect to the specifications of the initial design.

NO at worst-case operating conditions, which ensures that all specification parameters are met there, see the results of step 4 in Table 1. The runtime of NO without operating conditions is about 5 minutes, including operating conditions 15 minutes. Finally we employ YO to center the design in a pessimistic Monte Carlo model of the circuit from a simulated yield of 89% to almost 100%, in a runtime of less than 3 hours. In this step performances which impact yield are improved. The results after all these steps are also shown in Table 1.

Of course, the topology of our example low-power OTA is well-known and sizing rules can be found in the literature. Overall, the entire conversion, sizing for worst-case operating conditions and design centering of this OTA including verification required only very little effort from the designer (approximately a quarter of an hour excluding run time).

7 Conclusions

Circuit topology is the real analog IP, which is reused at different technology generations. The manual conversion of a complex circuit with up to 100 devices to another technology PDK is a time consuming task. Sizing the circuit costs a lot of time as well.

The presented method reflects common practice of analog design and is divided into design data conversion and sizing by optimization. Tools to support this flow are discussed.

Acknowledgment

The work described in this report is supported by the German Ministry of Education and Research (*Bundesministerium für Bildung und Forschung*) BMBF, grant ID 01 M 3086 F. The authors are responsible for the contents of this publication.

References

- [1] V. Boos, "Reuse of circuit topologies using WiCkED," in *MUGM '06: MunEDA User Group Meeting*, (Munich), 2006.
- [2] V. Boos, "Gaining insight into analog circuit behavior - tools and extensions to Cadence DFII," in *CDNLive! EMEA 2009*, (Munich), p. 66, 2009.
- [3] P. Schwarz, "Reuse of Mixed Analog-Digital Circuits," *it + ti*, pp. 91–98, 2002.
- [4] www.cadence.com.
- [5] www.muneda.com.
- [6] U. Sobe and U. Henniger, "zmdAnalyser: Ein Design Tool von Analog/Mixed Signal Designern," *DASS*, pp. 29–33, 2005.
- [7] Sherif Hammouda et al., "Chameleon ART: A non-optimization based analog design migration framework," in *DAC '06: Proceedings of the 43th annual conference on Design automation*, (New York, NY, USA), pp. 885–888, ACM, 2006.
- [8] H. Graeb, S. Zizala, J. Eckmueller, and K. Antreich, "The sizing rules method for analog integrated circuit design," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 343–349, 2001.
- [9] G. Stehr, M. Pronath, F. Schenkel, H. Graeb, and K. Antreich, "Initial sizing of analog integrated circuits by centering within topology-given implicit specifications," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2003.
- [10] K.-H. Rooch, U. Sobe, and M. Pronath, "Circuit design-for-yield (DFY) for a 110dB Op-Amp for automotive and sensor applications," in *GME/ITG-Diskussionsitzung Entwicklung von Analogschaltungen mit CAE-Methoden*, 2006.
- [11] R. Schwencker, F. Schenkel, H. Gräß, and K. Antreich, "The generalized boundary curve – A common method for automatic nominal design and design centering of analog circuits," in *Design, Automation and Test in Europe (DATE)*, pp. 42–47, Mar. 2000.
- [12] F. Schenkel, M. Pronath, S. Zizala, R. Schwencker, H. Graeb, and K. Antreich, "Mismatch analysis and direct yield optimization by spec-wise linearization and feasibility-guided search," in *ACM/IEEE Design Automation Conference (DAC)*, pp. 858–863, 2001.
- [13] K. J. Antreich and H. E. Graeb, "Circuit optimization driven by worst-case distances," in *The Best of ICCAD – 20 Years of Excellence in Computer-Aided Design*, pp. 585–595, Kluwer Academic Publishers, 2003.