

# gPCDS: An Interactive Tool for Creating Schematic Module Generators in Analog IC Design

Matthias Greif, Daniel Marolt, Juergen Scheible

Robert Bosch Center for Power Electronics, Reutlingen University  
Alteburgstrasse 150, 72762 Reutlingen, Germany  
{matthias.greif, daniel.marolt, juergen.scheible} @reutlingen-university.de

**Abstract**— Optimization-based design automation for analog ICs still remains behind the demands. A promising alternative is given by procedural approaches such as parameterized generators, also known as PCells. We are working on a complete analog design flow based on parameterized generators for entire circuits and corresponding layout modules. Because the conventional programming of such enhanced generators is far too complicated and costly, new methods are needed to ease their development. This paper presents gPCDS (graphical PCDS), a novel tool for a designer-oriented development of schematic module generators, integrated into a common schematic entry environment. The tool is based on PCDS (Parameterized Circuit Description Scheme), a meta-language for the creation of parametrized analog circuits. Schematic module generators are a very desirable complement to layout module generators in order to achieve a seamless schematic-driven layout design flow on module level. By facilitating a way of generator development that matches a design expert's mentality, gPCDS contributes to close this gap in the analog design flow.

**Keywords**— analog design automation; pcells; parameterized generators; procedures; analog synthesis; expert knowledge; re-use

## I. INTRODUCTION

The design of analog integrated circuits requires an extensive amount of expert knowledge and creativity, due to the big diversity of requirements to be considered (a difficulty known as “More than Moore”). Therefore, analog IC design is still dominated by manual work, which is time-consuming, costly, and error-prone. To improve this situation, two attempts come into consideration; (1) design automation and (2) design re-use.

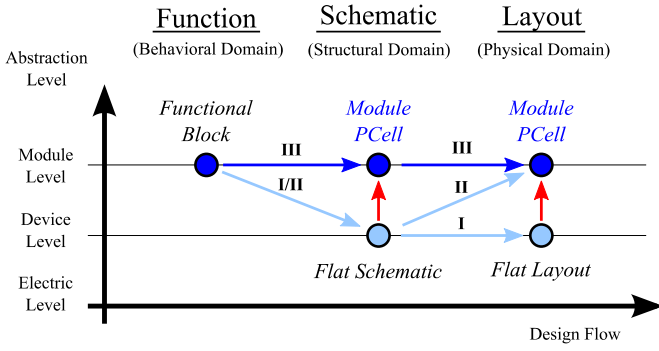
The design of an analog circuit involves the determination of a topology, followed by the sizing of the circuit. Both tasks are amenable to *design automation*, which is generally based on optimizing algorithms. The working principles of known approaches for the automatic creation of topologies are either synthesis or selection (from a given set of topologies). Tools for the synthesis of topologies can be divided into iterative and constructive algorithms. Iterative approaches determine an appropriate circuit structure by metamorphose of predefined architectural templates as demonstrated in [1]. Constructive approaches, as proposed in [2] and [3], assemble basic building blocks with a subsequent algorithmic exploration of the entire design space. Although commercial tools for topology synthesis have evolved, their industrial acceptance is rather limited.

In contrast to this observation, *design re-use* is common practice in industry, because existing designs are not only well-known within a design team, but also silicon-proven. Finished designs, however, cannot usually be taken over into new projects without any changes. In most cases, the circuits need at least a new sizing; but often the topology also has to be adapted, so in fact each re-use case adds a new circuit variant to a company's portfolio. This fashion of design re-use is driven by design projects rather than by a systematic strategy and leads to an uncontrolled -and unmaintainable- burst of design variants for every circuit class. It is obvious, that this re-use practice falls short of boosting the design productivity in a remarkable manner, especially when taking into account that often the physical design for all these variants has to be completely redone from scratch.

Our goal was to develop a holistic automation approach that elevates the entire IC design flow of analog circuits up to the *module level*. The approach is based on parametrized generators (PCells) for basic analog functions (e.g., OTAs, bandgaps, ADCs), denoted as *modules*. For each module, a schematic module generator (*SM-PCell*) is complemented by a corresponding layout module generator (*LM-PCell*). Thus, every module covers an entire circuit class, both on the schematic side and on the layout side.

Such a design flow on module level has four major benefits. (1) The implementation of entire circuit classes as parameterized generators enables a *systematic* design re-use. (2) Since generators replicate design experts' best practices, this represents not just a re-use of designs, but a re-use of *design strategies*, i.e., *design knowledge*. (3) The use of SM-PCells in conjunction with respective LM-PCells facilitates their integration in existing schematic-driven-layout (SDL) flows, and preserves all SDL functionalities. (4) Supporting each design step with module generators leads to a seamless flow on a higher level of design hierarchy (i.e., the module level), and this “bottom-up” automation paves the way for a successful “top-down” adoption of optimization-based algorithms, with the aim of a fully automated analog IC design flow as proposed in [5], where this vision is called a “bottom-up meets top-down” design flow.

This paper presents an essential component of our concept: gPCDS (graphical PCDS), an interactive tool that allows design experts to implement SM-PCells in an easy and designer-



**Figure 1.** Schematic-Driven-Layout (SDL) Design Flow; (I) Conventional SDL-Flow; (II) Design flow with layout module generator; (III) Seamless SDL-flow on module level

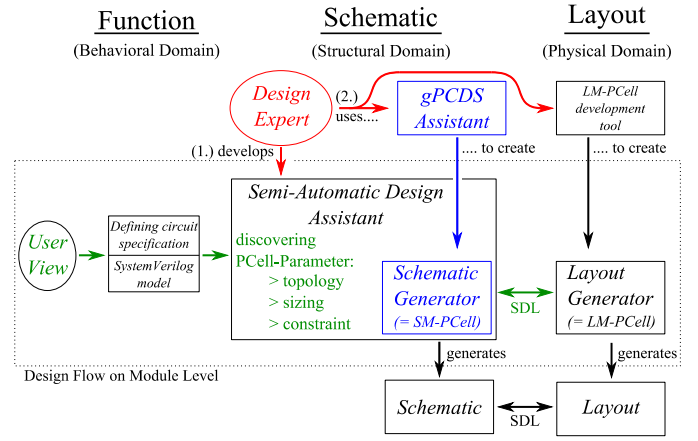
oriented manner. It is based on the *Parameterized Circuit Description Scheme* (PCDS) [6], a meta-language that simplifies the textual programming of SM-PCells. Enhancing PCDS, the novel gPCDS substitutes that non-intuitive code writing effort with a graphic-based way of SM-PCell creation. Thus, gPCDS enables to capture a design expert's knowledge implicitly in an easy way and translates it purely into a procedural description.

The paper is structured as follows. Section II discusses the characteristics of an analog IC design flow on module level in greater detail. Section III introduces the basic concepts of the existing PCDS approach. Section IV describes how gPCDS is implemented and provides some examples of its functionality. Section V concludes the paper with a summary and an outlook.

## II. A SEAMLESS DESIGN FLOW ON MODULE LEVEL

In IC design, three domains can be discerned [4]: (a) the behavioral domain, (b) the structural domain, and (c) the physical domain. These mirror the major design steps: (a) function specification, (b) schematic design, and (c) layout design. Figure 1 illustrates the common SDL flow throughout these steps and across the different levels of design hierarchy. As shown in flow I, circuit designers typically think of functional blocks (i.e., circuit classes), but the schematic and layout designs are created “flat” on device level. While LM-PCells (flow II) can increase the layout design productivity, the greatest benefit is certainly achieved when corresponding SM-PCells are introduced to resolve the hierarchy break (flow III). Here, it is obvious that SM-PCells are not only profitable for schematic design, but will also boost the development of powerful LM-PCells.

In addition to these assets, SM-PCells provide a solid basis for a successful application of algorithmic optimization where existing approaches fail to consider those low-level details that are implicitly taken care of by a PCell. Thus, our design flow includes a semi-automatic design assistant, as seen in Figure 2. To comprehend the philosophy behind the shown approach, one has to distinguish between a *user view* and an *expert view*. The differentiation is important since a pre-development work should be done by an experienced designer. This *Expert* develops the tools that are applied by the *User* afterwards.



**Figure 2.** Enhanced IC design flow with module generators; Horizontal: User view with semi-automatic design assistant, SM-PCell and appropriate LM-PCell; Vertical (Structural Domain): Creation of a schematic module generator by a design expert using the gPCDS tool.

### 1) User View (dashed rectangle)

The *user view* pertains to the automatisms that a design team can utilize in a design project. One automatism is the aforementioned design assistant, which is (again) a designer-oriented tool that bridges the function specification step with the schematic design. It provides an interactive topology selection and sizing mechanism, both of which are specifically dedicated to the class of the circuit that is currently under design. Given a set of performance parameters, the design assistant helps to convert these into input parameters for the respective SM-PCell, including the choice of topology and the device dimensions by using predefined test-benches. Provided that a corresponding LM-PCell is also available, the SDL flow ensures that the appropriate layout is automatically generated by the LM-PCell, which also get its set of parameters from the design assistant. In [7] LM-PCells, their parameters and a self-organization approach for layouts are explained more in detail.

### 2) Expert View

Elevating the design flow to the level of basic analog modules reflects the kind of abstraction that has been achieved in digital design decades ago: the basic design entities are logic gates - not transistors. However, while the digital design complexity is greatly reduced through the use of standard cells with fixed circuitry and normalized dimensions, such standardization is not feasible in the analog domain. This is because all degrees-of-freedom need to be exploited to meet the design requirements and yield the demanded level of quality. Since the design requirements strongly depend on the circuit's type, application and mission profile, and often also on the chosen semiconductor technology, it is inevitable that the module PCells –and also the associated functionality of the design assistant– are pre-developed by the respective design experts themselves. This pre-development work is depicted as the *expert view* in Figure 2. Concerning the realization of module PCells, a central objective is to yield the best possible trade-off between PCell re-usability and development effort. On the one hand, this requires a careful determination of the design space that is to be covered by the variability of the PCell. On

the other hand, a major challenge –but also a great opportunity– in this regard is to provide appropriate tools that allow design experts to convert their invaluable expert knowledge into procedural automatisms in an intuitive way. For SM-PCells, this implies meeting a circuit designer’s *mentality* – which is the driving force behind our graphical PCell development approach. Before discussing gPCDS in detail, Section III first gives a brief introduction to the PCDS approach, on which gPCDS is based.

### III. PARAMETERIZED CIRCUIT DESCRIPTION SCHEME

The generator development approach PCDS implements four major concepts that will be explained in this section.

#### A. Single Description – Dual Interpretation

A SM-PCell consists in fact of two PCells: a schematic PCell and a symbol PCell. In PCDS, both PCells are created from one single PCell description. A *PCDS Translator* converts the description into stand-alone low-level code of the design environment, which compiles the code into the two PCells. This not only reduces the development effort, but also helps in maintaining the consistency of the PCells.

#### B. High-Level PCDS Commands

PCDS provides high-level commands, most of which draw shapes (e.g., wires). Each command performs auxiliary background actions, such as binding the created shape to an internal identifier and storing specific geometric information of the shape. The purpose is to provide a syntactically compact notation style, which facilitates a *relative placement* technique.

#### C. Relative Placement and Generic Functionality

Relative placement is a key asset of PCDS by which new shapes are positioned in relation to previously created shapes. One great benefit of this technique is that changes to a shape position in a PCDS command call do not necessarily entail manual adjustments to the subsequent PCDS description. Addressing shapes and their members (e.g., points of a wire) is facilitated via the internal identifiers (see above) as well as *lexically* and *dynamically* created shape member designators. PCDS provides some generic features that are considered useful for any SM-PCell. One of these is *auto-flipping*: many basic analog transistor circuits have an NMOS and a PMOS variant with an equal topology; but the PMOS variant is usually drawn upside-down compared to the NMOS variant. The PCDS commands handle this automatically and eliminate the need for an explicit case distinction in the PCDS description.

## IV. THE gPCDS CONCEPT

#### A. gPCDS: Extension of Schematic Entry Tool

gPCDS can be seen as an extension of a commercial schematic entry tool, which remains the main operating window. The methods and functions of gPCDS are intuitive and user-friendly and furthermore, its handling does not require a familiarization process. Due to its popularity, we have chosen Cadence Virtuoso as basic design environment and SKILL as the native programming language. The GUI of gPCDS expands the functionality of the schematic editor by high-level functions for

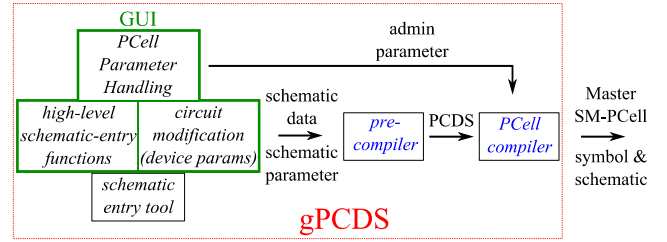


Figure 3. gPCDS data flow chart

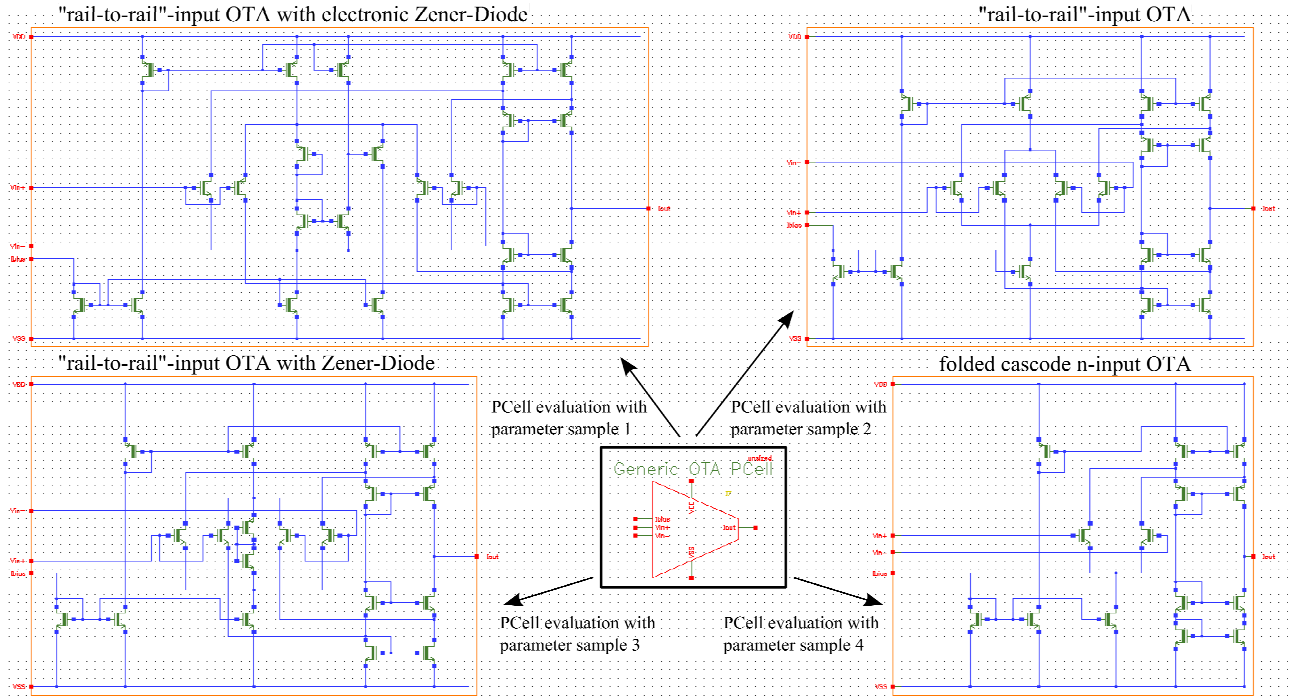
auto-connect and auto-route as well as an extendible building block library as proposed in [8]. To broaden the scope of application, a generic library is developed to achieve technology-independency. This enables a simple reuse and modification of the created schematic PCell after transferring it to other technologies.

#### B. gPCDS: Parameterization Concept

According to the PCell parameters (c.f. Figure 2) we differentiate between *schematic parameters* and *admin parameters*. The schematic parameters face the circuit variances, which are in fact incremental changes in sizing and topology as well as their conditions of occurrence. In addition, sizing dependencies (e.g. current mirror ratio) and rules (e.g. matching constraints) can be defined by the schematic parameters. These parameters enable the design expert to attach layout constraints to the SM-PCell. The SM-PCell, in turn, passes this physical design information to the corresponding LM-PCell. Besides the parameters concerning the schematic, gPCDS enables the adoption of administration parameters that effect the entire module PCell. These high-level parameters cover, amongst others, a connectivity-highlighting for the SM-PCell circuitry, or a topology selection out of a number of topologies. To enable a simple database handling, all information needed for the source code of the SM-PCell is embedded in the data structure of the used design environment. While the design expert can attach the schematic parameters directly into the schematic view, the admin parameters are defined inside the GUI of gPCDS and are finally stored into the PCell’s procedural description during the compilation of the PCell. Both the schematic parameters as well as the admin parameters are directly interpreted and evaluated during the instantiation of the SM-PCell.

#### C. gPCDS: Data Compilation

Figure 3 pictures the data flow chart for an SM-PCell generation with gPCDS. As depicted there, gPCDS acts as a dual compiler that reads (decompiles) the schematic netlist data and the schematic parameters from the schematic view, generates a description in the meta-language PCDS and finally compiles this PCDS-code into a PCell description in the design environment’s native programming language. The pre-compiler thereby accesses the *open access* database from Virtuoso, reads the embedded data and converts it to PCDS-code. We insist on PCDS as an intermediate step, due to its characteristics mentioned in Chapter III and to guarantee the technology independency and thus the knowledge transferability to other design environments. The data transfer from the open access database into PCDS-code can be done for every arbitrary schematic view, even for old projects. To encourage the reuse of already existing designs, gPCDS provides a “merging” func-



**Figure 4.** A demonstration of a schematic module PCell for the OTA circuit class and its instances according to the chosen parameters.

tion to concatenate several schematic views (circuits) into a single PCell. This enables a parameter driven selection from a pool of topologies, which is often a simpler solution than generating all desired topology variants by parameter driven modifications of one basic topology. This functionality eases the achievement of a desired coverage of topologies within a certain circuit class. Figure 4 presents an SM-PCell example, implemented with the merging function, for the OTA (Operational Transconductance Amplifier) circuit class according to the findings of [9]. To simplify the form of representation we reduced the symbol view of the SM-PCell by an appropriate OTA-symbol as it is commonly used. Beside the OTA-symbol (lower middle of Figure 4), four instantiated examples according to a particular set of input parameters are depicted.

## V. CONCLUSION AND OUTLOOK

This paper presents gPCDS, an interactive tool for creating analog schematic module generators. gPCDS supports a convenient PCell creation flow, which is kept close to the designer's familiar way of creating schematics. Thus, gPCDS greatly facilitates the reuse of circuit design knowledge implicitly included in generators. Because no additional code writing is required, this designer-oriented approach is widely accepted by design experts. The generator approach is able to cover the entire variability of a typical analog circuit class with a size of up to about 50 devices with sufficient flexibility in order to achieve best quality results. Future works will concentrate on the semi-automated assistant and the development of a corresponding tool for its creation.

## ACKNOWLEDGMENT

This project is funded by the German Federal Ministry for Education and Research under the support code: 03FH03413.

## REFERENCES

- [1] Wim Kruiskamp, Domine Leenaerts. "DARWIN: CMOS opamp synthesis by means of a genetic algorithm". In *Design Automation, 1995. DAC '95. 32nd Conference on*, vol., no., pp.433-438, 1995. DOI= 10.1109/DAC.1995.249986
- [2] Mitea, O.; Meissner, M.; Hedrich, L.; Jores, P. "Automated constraint-driven topology synthesis for analog circuits". In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*, vol., no., pp.1-4, 14-18 March 2011. DOI= 10.1109/DATE.2011.5763264
- [3] Meissner, M.; Hedrich, L. "FEATS: Framework for explorative analog topology synthesis". In *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol.34, no.2, pp.213-226, Feb. 2015. DOI= 10.1109/TCAD.2014.2376987
- [4] Gajski, D.D.; Kuhn, R.H. "Guest Editors' Introduction: New VLSI Tools". In *Computer*, vol.16, no.12, pp.11,14, Dec. 1983. DOI= 10.1109/MC.1983.1654264c
- [5] Scheible, J.; Lienig, J. "Automation of Analog IC Layout – Challenges and Solutions". In *Physical Design (ISPD), 2015, Proc. of the Int. Symposium on*, pp.33-40, DOI = 10.1145/2717764.2717781
- [6] Marolt, D.; Greif, M.; Scheible, J.; Jerke, G. "PCDS: A new approach for the development of circuit generators in analog IC design". In *Microelectronics (Austrochip), 22nd Austrian Workshop on*, vol., no., pp.1.6, 9-9 Oct. 2014. DOI= 10.1109/Austrochip.2014.6946310
- [7] D. Marolt, J. Scheible, G. Jerke, V. Marolt, "SWARM: A self-organization approach for layout automation in analog IC design," *Int. Journal of Electronics and Electrical Engineering (IJEEE)*, 2016, Vol. 4 (5), pp. 374-385.
- [8] Massier, T.; Graeb, H.; Schlichtmann, U. "The Sizing Rules Method for CMOS and Bipolar Analog Integrated Circuit Synthesis". In *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol.27, no.12, pp.2209-2222, Dec. 2008. DOI= 10.1109/TCAD.2008.2006143
- [9] A. Gerlach, J. Scheible, T. Rosahl and F. T. Eitrich, "A generic topology selection method for analog circuits demonstrated on the OTA example," *Ph.D. Research in Microelectronics and Electronics (PRIME), 2015 11th Conference on*, Glasgow, 2015, pp. 77-80. doi: 10.1109/PRIME.2015.7251338