1 Introduction

In the semiconductor industry one of the key factors for staying competitive is to continuously improve efficiency. In 1965, Gordon E. Moore predicted a rapid exponential growth of the number of transistors that can be integrated on a chip. His prediction became commonly known as “Moore’s Law”. Since then, the semiconductor industry’s growth rates have proven Moore’s early extrapolation. The semiconductor market continuously creates demand for more complex integrated circuits at ever lower cost. This trend pushes the semiconductor companies to improve their efficiency in all fields. Cost efficiency in production is achieved by shrinking transistor sizes and increasing productivity of the semiconductor fabs. Higher system integration leads to complex systems being integrated on a single chip in order to improve the assembly cost for customers. Last but not least, the design efficiency has to keep track with the rapid development of technology. Resources to develop a chip are limited and the project cycle times play a crucial role in timely bringing profitable products to the market. Thus, the lever to become more efficient within a chip’s design phase are design methodology and design automation.

Design efficiency is mostly driven by Electronic Design Automation (EDA). The EDA industry is closely connected to the semiconductor companies as the fulfillment of Moore’s law would not have been possible without a large degree of automation in chip design. The design is based on a design flow, which is composed of a large variety of specialized tools to support the process steps from specification to production. While the design of digital systems is largely automated and dominated by a clearly defined design methodology, analog designs are mostly left to the experience of specialized design engineers. Even though the analog subsystems of a chip are increasingly becoming more important, analog design is insufficiently supported by automated design tools. New design methodologies are necessary to improve efficiency and prevent costly redesigns due to the late detection of errors.

Design and Verification of Microelectronic Circuits

Traditionally, mixed-signal design was performed by bottom-up design. Starting from the design and verification of individual circuit blocks, the obtained components were integrated into the system and verified at transistor level. This design methodology posed several problems such as high simulation effort, disadvantages for architectural changes, risk of communication errors, and late recognition of errors. In order to tackle these problems, top-down design methodologies are increasingly applied [41, 42]. They enhance the efficiency and quality of the design process due to their well-structured refinement from an architecture to a transistor level realization. Each level is thoroughly partitioned, designed, and refined to

the next level followed by a verification step. Thereby, the system is step-wise designed from an algorithmic description at the system level down to a transistor level realization of all blocks.

Verification is the process of proving the compliance of a circuit with its specification. For microelectronic systems, verification is of major importance as there is no possibility for prototyping, and redesigning after the production start causes enormous costs. The most common verification method is circuit simulation, which – in a strict sense – does not prove but only validate the circuit. Circuit simulation tools are intended to numerically predict the behavior of a circuit's electrical quantities without having an actual realization of it. They are based on parameterized device models to describe the behavior of the basic electrical components of the circuit. Netlists are used to list the circuit's components and describe their interconnecting network. The simulation of electrical systems can be performed at different abstraction levels:

- **Digital simulation** – time- and value-discrete simulation method based on boolean logic that is capable of simulating large digital circuits with considerable computing resources (millions of transistors within a day).
- **Analog simulation** – continuous value simulation method with adaptive time steps for analog circuits that yields accurate results for currents and voltages, strongly restricted by computing power (thousands of transistors within a day).
- **Mixed-signal simulation** – a combination of the previously mentioned methods that adaptively uses one of the methods for digital or analog partitions of the circuit (hundred-thousands of transistors within a day).
- **Device simulation** – highly accurate field solver to calculate physical behavior of a single or very few semiconductor devices that requires large amounts of computing power (few transistors within weeks, inappropriate for circuit simulation).

The examples for the simulation time give a rough idea of the typical capability of each simulation type. A comparison of the simulation methods shows that the accuracy of the simulation results and the necessary simulation times are conflicting interests. The term accuracy within the simulation-context specifies the degree of conformity of the calculated to the measured values. In order to achieve a high accuracy within simulation, new devices as well as new technologies require the characterization of the devices to achieve suitable parameter sets for the corresponding simulation models. The determination of the device parameters is based on measured characteristics to calibrate the device models' behavior. Due to limited computing resources, performance is often the limiting factor that requires the application of less accurate simulation methods. For circuit simulation, analog simulation is considered to be the most accurate and feasible solution. Device simulators are not suited for circuit simulation due to the required amount of computing power, even though they would be more accurate.

Figure 1.1 visualizes the relationship between accuracy and performance of a simulation. Considering computing resources and efficiencies as constants, an increased accuracy of the models used within the simulation proportionally increases the simulation effort. Thereby, the simulation time increases and performance is affected. The only possibility to enhance

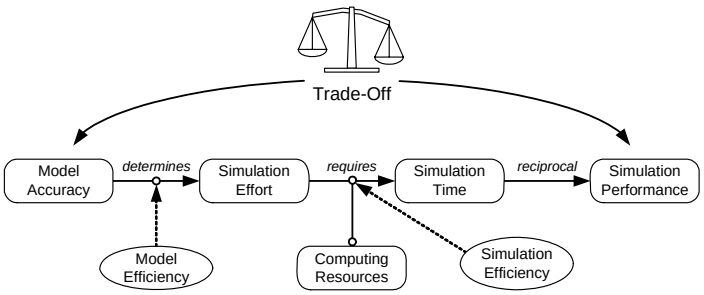

Figure 1.1: Accuracy Performance Trade-Off

performance without loosing accuracy is to increase computing resources (as e.g. applied in parallel computing) or to improve efficiency of the model or the simulator. The scaling of the simulation effort with the model accuracy is influenced by the model efficiency, which is determined by the realization and formulation of the model. The efficiency of the simulator determines the necessary simulation time for a defined simulation effort with a given amount of computing resources. Depending on the application, a suitable trade-off between accuracy and simulation time has to be found. In order to make this trade-off as profitable as possible, the efficiency of the model as well as the simulator must be optimized. Enhancing simulation efficiency is typically impossible for the user of a simulator. Improving the model's efficiency is possible for the creator of the model but requires some internal knowledge of the simulation algorithms and should ideally be done automatically by the simulation environment.

Performance-wise, the verification of large mixed-signal systems on chip level is the most crucial issue in circuit verification. Simulating the entire chip with analog accuracy is almost always not a feasible solution due to the extremely high computational effort. Using a digital simulator is impossible due to the analog subsystems of the chip that cannot be simulated with digital simulation algorithms. In most cases, even the application of a mixed-signal simulator does not reduce the computation time to target (typically over-night simulation).

Behavioral Modeling

The use of behavioral models is a strategy to speed-up simulations. It becomes increasingly important for top-down as well a bottom-up design methodologies. A behavioral model is a functional description of a specific circuit that is suited to predict the relevant behavior of the corresponding circuit with reduced simulation effort. According to Figure 1.1, this reduction of the simulation effort comes along with reduced accuracy of the simulation results. Typically, this is achieved by neglecting physical effects of the circuit implementation that are considered irrelevant for the application of the model. Subsequently, the behavioral model can be used to replace its circuit-counterpart in order to speed-up the verification in larger contexts. The strategy to simulate a system partly represented by its circuit netlist and partly by behavioral models is called multi-level or mixed-mode simulation [11]. By simulating

varying combinations of circuits and behavioral models, the functionality of the whole system or of specific components within the system context can be verified.

The most common types of behavioral models are:

- **Electrically equivalent circuits** (macro models) – simplified circuits to model the terminal characteristics of the original circuit [9, 10, 67]. Historically, this is the first approach of behavioral modeling as the models can be simulated with an ordinary circuit simulator.
- **Equation-based models** – behavioral models based on mathematical systems which are typically realized in an Analog Hardware Description Language (AHDL) and require a corresponding simulator interface [4, 33, 48].
- **Look-up tables** – behavioral models based on sampling points stored within data tables. In conjunction with an interpolation method, it is possible to "look up" output characteristics of the model dependent on the input values [78, 86]. This model type is desirable for modeling applications where no equation-based description is available.

For detailed comparisons and discussions on modeling approaches and classifications of behavioral models please refer to [3, 58, 66]. Modeling approaches can be classified into empirical and analytical methods. The former only uses observations, e.g. measurements or simulation data, to reproduce a circuit's behavior. This has disadvantages as the model does not reflect physical properties of the circuit. Analytical modeling methods are based on physical laws and interrelationships of the modeled circuit. Therefore, a precise analysis and understanding of the circuit is necessary. Analytical modeling methods are superior to empirical methods as they provide insight into the model's behavior and offer the possibility of adapting the model to circuit changes. Analytical models are equation-based, but not all equation-based models are analytical.

As manual modeling is time-consuming, error-prone, and requires a high level of modeling knowledge, an automated modeling technique is highly desirable. Especially for bottom-up modeling with the intention of deriving a behavioral model from an already implemented circuit block, several automated modeling approaches exist:

- **Characterization** – a library of parameterized model templates allows modeling of specific circuit classes. The parameters for the selected model are determined by characterization of the circuit [19, 37].
- **Neural networks** – behavioral models based on neural networks that are trained with simulation or measurement data [17, 52, 53].
- **Symbolic analysis** – an approach to generate equation-based models using a computer algebra system in combination with network analysis algorithms [3, 5, 27, 31, 65, 87].

This work focuses on automated bottom-up generation of equation-based behavioral models for nonlinear analog circuit blocks through symbolic analysis as introduced in [3]. The approach is based on the automated derivation of symbolic network equations from a circuit within a computer algebra system. The core of a symbolic analysis system is its model reduction algorithm – the process of simplifying equations until a user-specified accuracy-criterion is reached. This method is very useful for bottom-up modeling as it approximates the

circuit with its own network equations. The resulting simplified set of equations can be used as core of an equation-based behavioral model. The suggested modeling method has several advantages over other approaches:

- Automated modeling process
- Model accuracy specified in advance
- Very high accuracy attainable with limited modeling effort
- Applicable to all circuit classes (limited to analog block size)
- Resulting models parameterized with dominant circuit parameters
- Insight into the model equations

Chapter 2 will discuss symbolic analysis and its application for behavioral modeling in more detail. An introduction to the relevant simulation algorithms for nonlinear dynamic systems and to behavioral simulation methods will follow in Chapter 3.

Motivation

Even though highly efficient model reduction techniques exist, the generated behavioral models contain equation systems of exceptionally high complexity. Unfortunately, the simulation performance of the generated models is often significantly lower than the performance of the corresponding netlist-based simulation, making their use impossible. Example 1.1 illustrates this problem.

Example 1.1: Performance Problem

In [100], the behavioral model generation for a complementary folded-cascode operational amplifier was published. The operational amplifier consists of 19 MOS-transistors (modeled with BSIM3v3 [89]). This analog block was intended to be modeled through symbolic analysis to achieve a behavioral model with a 10 % error bound of the amplifier's output voltage. Initially, the equation setup resulted in a complex equation system of 1177 equations – with the majority being highly nonlinear. Through automated model reduction, the equation system was reduced to 29 equations only – still fulfilling the required error margin. The simulation time for the generated behavioral model was enhanced by a factor of 16. Still, the simplified model's simulation performance was 4 times worse than the performance achieved through the netlist-based simulation of the original circuit.

■

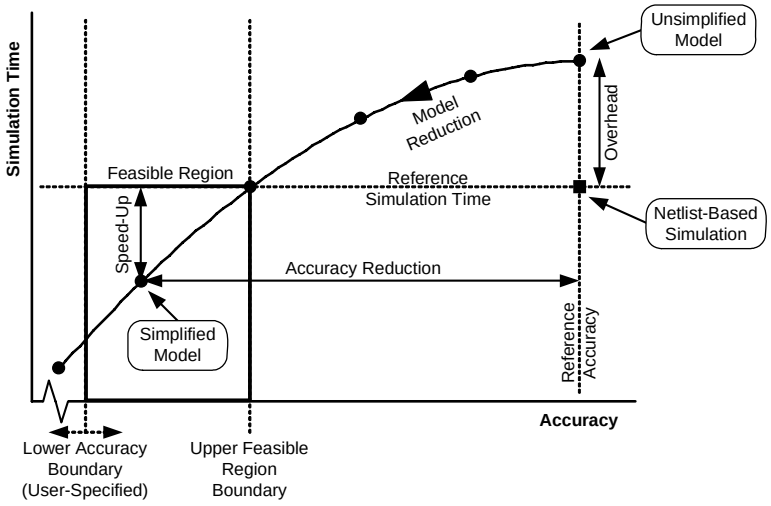

Figure 1.2: Simulation Time vs. Accuracy for Modeling through Symbolic Analysis

Figure 1.2 qualitatively depicts the current situation with respect to the accuracy performance trade-off for this modeling approach. The reference in terms of accuracy and simulation time for all bottom-up modeling methods is the netlist-based simulation (black square in Figure 1.2), as the model is intended to replace (and speed-up) this representation of the circuit block. It is supposed to be the most accurate simulation type for the modeled circuit block. Based on the netlist, symbolic analysis offers the possibility to generate a behavioral model containing equivalent network equations as used simulator-internally for the netlist-based simulation (unsimplified model). This model has the same accuracy as the netlist-based simulation but typically a significantly higher simulation effort, resulting in an overhead in simulation time. Starting from this unsimplified model, a plurality of simplified models along a decreasing trajectory can be achieved through model reduction. The trajectory reflects the trade-off between accuracy and simulation time for different degrees of model reduction. The shape of the curve was chosen exemplarily. In practice, it highly depends on the structure of the model equations and the applied model reduction algorithms. Hence, the trajectory may be of arbitrary shape but should be monotonic decreasing.

The feasible region for a practically useful model is limited by the user-specified minimum accuracy and the requirement to speed-up the simulation (upper feasible region boundary). A simplified model close to the lower accuracy boundary minimizes simulation time. Due to the reduction in accuracy, a certain speed-up compared to the reference simulation time is achieved. In the case of Example 1.1, no feasible compromise between simulation time and accuracy could be found as the upper boundary did not comply with the accuracy require-

ments. A certain amount of accuracy reduction is necessary to compensate for the overhead in simulation time and to speed-up the model to reach the reference simulation time. Thereby, the efficiency of the modeling approach significantly degrades – this amount of accuracy reduction is "wasted" without achieving a speed-up compared to the reference simulation.

Objectives of this Work

Within Chapter 4, analyses with respect to the behavioral models' simulation performance will be presented. They show that the overhead is far from being negligible – in most cases the unsimplified model is in the order of one to two magnitudes slower than the netlist-based simulation. The main objectives of the performance analyses are the quantification of the overhead, the investigation for the root causes of the inefficiency, and the determination of influencing factors that account for the overhead.

Certainly, further reducing the model's accuracy to compensate for the initially bad performance of the unsimplified models is not a satisfying solution. The main objective of this work is to maximize performance through efficiency improvements of both the models and the simulation process. The overhead should be reduced to a minimum in order to make this modeling approach competitive in terms of simulation performance and to effectively use the powerful model reduction algorithms for speeding-up the behavioral models compared to the netlist-based simulation.

Based on the results of the performance analyses, Chapter 5 will present approaches to enhance the behavioral simulation efficiency. Automated optimization methods to increase the model efficiency by reducing the simulation effort at constant accuracy are presented in Chapter 6. Both measures are strongly related to each other as optimal efficiency requires an adaptation between the behavioral model structure as well as the applied simulation algorithms.