Technische Universität München
Fakultät für Elektrotechnik und Informationstechnik

Functional Block-Based Synthesis and Sizing of Integrated Operational Amplifiers

Inga Abel

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik zur Erlangung des akademischen Grades einer

Doktorin der **Ingenieur**wissenschaften **(Dr.-Ing.)**

genehmigten Dissertation.

Vorsitzender:	Prof. Dr.-Ing. Werner Hemmert
Prüfer der Dissertation:	1. Apl. Prof. Dr.-Ing. habil. Helmut Gräb
	2. Prof. Dr.-Ing. Lars Hedrich

Die Dissertation wurde am 14.12.2021 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 13.05.2022 angenommen.

Abstract

This thesis presents a new computer-oriented modeling for analog circuits and applies it to circuit synthesis and sizing. Op-amps are modeled by hierarchical functional blocks having a unique functional and structural description. Algorithms are presented to identify the functional blocks in op-amps automatically. For each functional block, an analytical performance model is derived and stored in a library. Based on this hierarchical performance equation library (HPEL), a topology independent automatic sizing method is developed emulating the manual design process. A structural synthesis method uses the functional block modeling of analog circuits to create thousands of meaningful op-amps and evaluates them for given specifications. The methods and algorithms developed in this thesis support around 4000 different topologies. Experiment results show the effectiveness of the approaches.

Zusammenfassung

Diese Arbeit präsentiert eine neue computerorientierte Modellierung für analoge Schaltungen und wendet diese auf Struktursynthese und Schaltungsdimensionierung an. Operationsverstärker werden durch hierarchische Funktionsblöcke modelliert. Jedem Block wird eine eindeutige Funktions- und Strukturbeschreibung zugeordnet. Algorithmen zur automatischen Klassifizierung von Funktionsblöcken anhand der Beschreibungen werden präsentiert. Für jeden Funktionsblock wird ein analytisches Verhaltensmodell entwickelt. Eine neue Dimensionierungsmethode nutzt diese Verhaltensmodelle, um Operationsverstärker topologieunabhängig zu dimensionieren. Die Methode emuliert den manuellen Entwurfsprozess. Eine Struktursynthesemethode nutzt die Funktionsblockmodellierung von Operationsverstärkern, um tausende praktikable Topologien zu erstellen und mit Hilfe der Dimensionierungsmethode zu evaluieren. Die in dieser Arbeit entwickelten Methoden und Algorithmen unterstützen ca. 4000 verschiedene Operationsverstärkertopologien. Experimentelle Ergebnisse bestätigen die Effektivität der Methoden.

Preface

Ever since I started to work at the Chair of Electronic Design Automation, I wanted to work on the structural synthesis of analog circuits. Early, I learned that there were many obstacles to take and only a few attempts have been made on that problem so far. I realized that my chances to work on structural synthesis were very small. However, by writing my master thesis about automatic sizing of operational amplifiers and having the chance to spend my first year of my doctorate studies solely on enhancing the algorithm in such a way that it supports many topologies, I could finally reach this goal and wrote a structural synthesis algorithm for analog operational amplifiers based on the previous developed sizing method.

This, however, would not have been possible without the funding of the Cusanuswerk. Therefore, I would like to thank the Cusanuswerk for granting me the doctorate scholarship and supporting me along my journey. I also would like to thank the head of the chair of Electronic Design Automation, Prof. Dr.-Ing. Ulf Schlichtmann, for letting me use the equipment and rooms of the chair and also providing me additional funding so that living in Munich became more affordable. A special thanks to Prof. Dr.-Ing. Helmut Graeb for being my advisor, helping me throughout my studies, papers, and the many requests for the extension of my scholarship. I also want to express my gratitude to the second reviewer of this work for his interest in the thesis and for the time he spent to review it.

I thank my mentor, Alexander Stanitzki, from Fraunhofer IMS, for all his support, especially, for the expertise he provided in analog circuit design. I also thank Petra Wendt for the time she spent correcting my linguistic mistakes in the thesis.

I would like thank all my former colleagues at the chair, especially, Maximilian Neuner, for advising me in the beginning of my studies on electronic design automation of analog circuits and later on for all our fruitful discussion during our pre-corona coffee breaks. I also would like to thank all the students who helped to improve the algorithms.

And finally, I would like to thank all my friends and family for all their support and also for making life much easier to live.

Munich, November 2021

Contents

1. Introduction

Analog and mixed-signal circuits play an important role in modern electronic devices. They are part of many integrated circuits (ICs) components with a growing importance in industry production. The market growth of integrated circuits emerged from $10 billion in 1980 to more than $340 billion in 2015 ("World Semiconductor Trade Statistic," https://www.wsts.org/, [1]). The market growth of analog circuits increased even faster with 10.6% in 2014, being 0.7 points over the overall market growth [1]. Taking the numbers from the *World Semiconductor Trade Statistic*, it is without question that analog circuits still play an important part in modern ICs. However, their design comes along with some obstacles. One is the nature of analog components. Even if parts of previously designed circuits are reused, an analog component always has to be adopted to the specific use-case. Another is the analog design process. According to the *International Technology Roadmap for Semiconductors* [2], the automation of the analog design process lags behind the digital one. While the digital design process is automated on a high level of abstractness - functions of a circuit can be programmed in VHDL or SystemC, layout and hardware are nearly automatically generated -, the computer-aided design (CAD) tools for analog circuits mostly support the designer with realistic simulations of the circuit. Schematics with device dimensions and the corresponding layouts have to be designed manually in the software environment.

According to a study of the IBS Group cited in 2005 in EDA Weekly, on a common IC, average 20% of the chip area is analog. However, 40% of the design effort and 50% of the design revision are due to analog components [3]. According to *Graeb, 2012* [4], the numbers might still be valid as they correspond to the still practiced manual design process of analog circuit. Thus, an automation of analog design process is still needed to reduce the error-proneness of the design and the overall design time.

Fig. 1.1 gives an overview of the classical analog design process used in industry. For given performance requirements, e.g., maximum gain and slew rate, a topology is selected. In the next step, the designer estimates adequate component dimensions, so that the topology fulfills the performance requirements. The sizings of the topology components as, e.g., transistors, resistors and capacitors, are calculated using analytical equations describing these performance features. The equations are topology dependent, i.e., they must be derived for every individual topology. An overview of a topology and its analytical performance equations is given in Fig. 1.2. A detailed description of the equations is given in Sec. 1.1. The calculated component dimensions are evaluated by numerical simulations using CAD

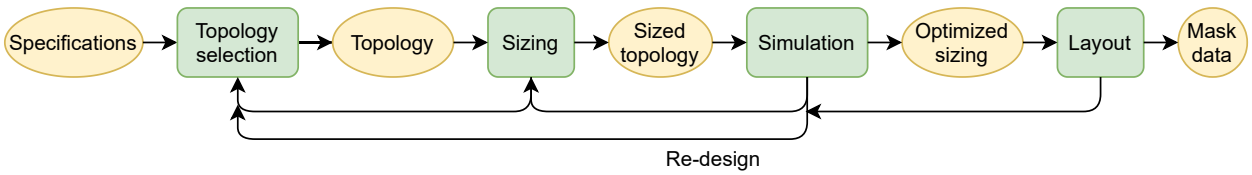
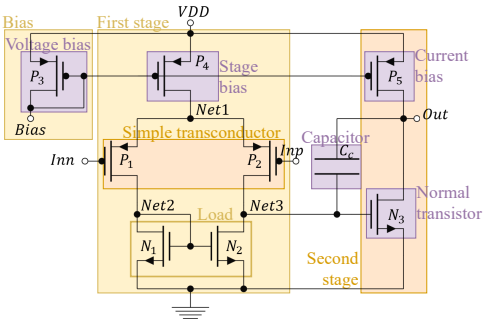


Figure 1.1.: Classical analog design process



(a) Miller op-amp

Performance features	Analytical performance equations
Slew rate	$min\{\frac{i_{DS,P_5}}{C_L}, \frac{i_{DS,P_4}}{C_C}\}$
Open-loop gain	$\frac{gm_{P_1}}{gd_{P_1}+gd_{N_1}} \cdot \frac{gm_{N_3}}{gd_{N_3}+gd_{P_5}}$
Quiescent power	$VDD \cdot (\lvert i_{DS,P_3}\rvert + \lvert i_{DS,P_4}\rvert + \lvert i_{DS,P_5}\rvert)$
Unity-gain bandwidth	$\frac{gm_{P_1}}{2\pi C_C}$

(b) Performance features and their analytical model equations

Figure 1.2.: Example topology and its analytical performance model

tools as, e.g., Cadence Virtuoso [5]. If the specifications are not fulfilled by simulation, the component dimensions are optimized. If there are great differences between calculated and obtained performance values, the topology might also be redesigned. If the simulated values are satisfactory, it is proceeded with the layout of the circuit, from which the mask data is obtained. With the mask data, the circuit is manufactured.

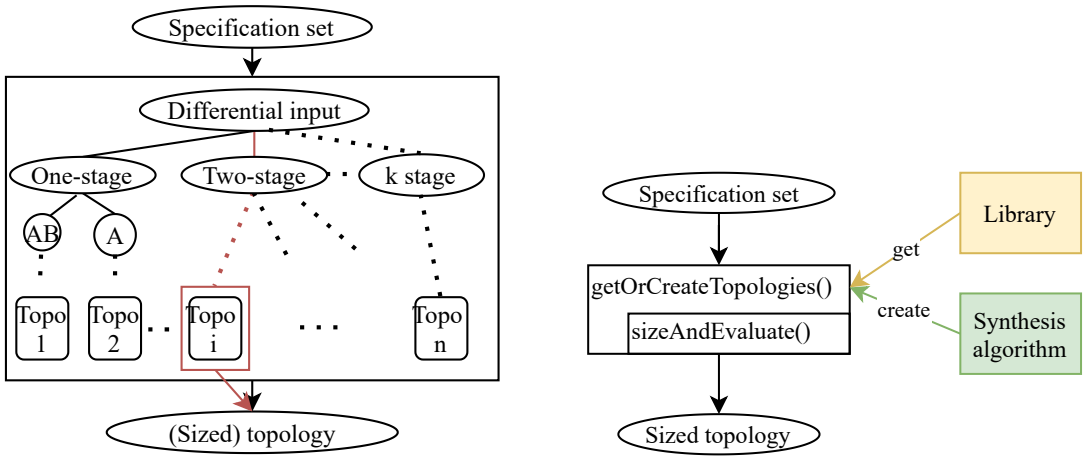
In research, different approaches exist to automate the classical design process of analog circuits (Fig. 1.1). An overview of the different tools is given in [6]. They can be classified as follows:

- Automatic sizing of analog circuits, e.g., [7–26]
- Automatic layout design, e.g., [27–33]
- Structural synthesis of analog circuit based on given specifications, e.g., [34–46]

This work focuses on automatic sizing and structural synthesis of analog circuits. Methods to automate the topology selection process and the sizing process of analog circuits are presented. We will consider one group of analog circuits in particular - operational amplifiers. Operational amplifiers are fundamental building blocks of analog/mixed-signal circuits. They mean to analog design, as some say, what inverters mean to digital design. As the industrial standard in op-amp design is equation-based, we emulate this equation-based process and make it computational useable. This is different to the state of the art in automatic sizing which mainly uses numerical methods [16–25]. In the following section, the steps are described, which are needed to automate the manual topology selection and sizing process of operational amplifiers.

1.1. Automating the Manual Topology Selection and Sizing Process of Op-Amps

Compared to other analog circuit classes, the design process of op-amps is well described in literature [47–51]. For many different topologies, their equation set-up for sizing is described in detail. Recommendations are given which topologies are suited for which type of specifications.



(a) Knowledge-based topology selection, e.g., [42]

(b) Topology selection through sizing, e.g., [36, 45]

Figure 1.3.: Two approaches to automate the topology selection process

1.1.1. Automating the Topology Selection Process

To automate the topology selection process, the knowledge of the behavior of a topology must be made computational accessible. Two types of approaches can be distinguished (Fig. 1.3):

- Designer knowledge of the behavior of different topologies is used in form of if-then-else decision-making that follows fixed paths to decide for a topology (Fig. 1.3a), e.g., [38, 42].
- Different topologies are evaluated by trying to find component dimensions for the topologies fulfilling a set of given specifications (Fig. 1.3b), e.g., [40, 43, 45].

The topologies in both approaches are either part of a fixed library (library-based) or are generated based on a library of building blocks (synthesis-based). Topology synthesis is in particular suitable for the evaluation process based on sizing (Fig. 1.3b), as a topology independent sizing process can evaluate many different topologies. In the next section, automatic sizing approaches are described suitable to automate the evaluation process based on sizing. In the knowledge-based selection process (Fig. 1.3a), the topology alternatives are adapted to the decision-making in the process. If a new topology is added, the selection paths must be adjusted to the topology. Thus, these approaches are mostly library-based [37, 38, 42]. The disadvantage of library-based approaches using either knowledge-based decision-making (Fig. 1.3a, e.g., [37, 38, 42]) or topology evaluation through sizing (Fig. 1.3b, [36]) for topology selection is that they only support a small number of topologies. Topology synthesis (Fig. 1.3b, e.g., [40, 43–45]), however, might create redundant or impractical topologies, which must be discarded during the sizing process.

A topology selection method is needed which allows a fast integration of new topologies in the process, supports a high number of topologies, but excludes impractical and redundant topologies from the process. This can only be achieved by a method which is as topology independent as possible but sticks close to the manual design process. The topology independence allows a fast inclusion of new topologies in the process. The closeness to the analog design process omits redundant and impractical topologies.

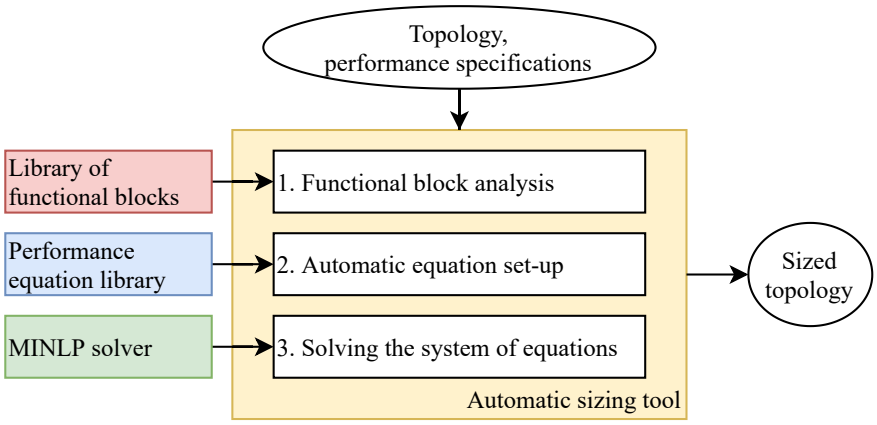


Figure 1.4.: Automating the manual sizing process

1.1.2. Automating the Manual Sizing Process

In the manual sizing process, analytical equations are used to size a specific topology. Fig. 1.2 shows some of these equations used to find the dimensions of the components defining a simple Miller op-amp. The equations describe the performance features $\mathbf{z}$ of the op-amp, e.g., open-loop gain $z_{A_{D0}}$, slew rate z_{SR}, with $\mathbf{z}$ being a N_z-dimensional vector. The variables of the equations are the drain-source current i_{DS,t_k}, the transconductance gm_{t_k} and output conductance gd_{t_k} of all transistors $t_k \in T$ in the circuit. Using a transistor model, $i_{DS,t_k}, gm_{t_k}, gd_{t_k}$ can be written as a function of the transistor dimensions w_{t_k}, l_{t_k} and other transistor variables as drain-source and gate-source voltage v_{DS,t_k}, v_{GS,t_k}:

$$i_{DS,t_k} = f_{i_{DS,t_k}}(w_{t_k}, l_{t_k}, v_{DS,t_k},\ v_{GS,t_k}),\ gm_{t_k} = \frac{\partial i_{DS,t_k}}{\partial v_{GS,t_k}},\ gd_{t_k} = \frac{\partial i_{DS,t_k}}{\partial v_{DS,t_k}} \tag{1.1}$$

For a given set of specifications $\mathbf{s}_z$, the performance equations written as a functions of the device dimensions $\mathbf{f}_z(\mathbf{w}, \mathbf{l})$ are solved for the widths $\mathbf{w}$ and lengths $\mathbf{l}$ of all transistors in the circuit being a non-linear system of equations:

$$\mathbf{s}_z = \mathbf{f}_z(\mathbf{w}, \mathbf{l}) \tag{1.2}$$

In this equation, $\mathbf{w}, \mathbf{l}$ are $|T|$-dimensional vectors. $\mathbf{s}_z, \mathbf{f}_z(\mathbf{w}, \mathbf{l})$ are N_z-dimensional vectors. In the manual design process, a system of equations determines the device dimensions. Please note that hereinafter all column vectors are written in bold lowercase letters.

To automate the design process, the non-linear equation system must be automatically set-up and solved. The performance equations describing the behavior of a topology as, e.g., in Fig. 1.2b, and additional symmetry and performance constraints must be automatically instantiated. Fig. 1.4 gives an overview of the three steps needed to automate this process.

To automate the set-up of the equation system, the function of each transistor in the circuit must be known. Based on this information, transistor variables as $i_{DS,t_k}, gm_{t_k}, gd_{t_k}$ of specified transistors are selected to serve as input to the performance equations (e.g., Fig. 1.2b). Analog designers know the function of a transistor by inspection. They know which transistors form the first and second stage, which the circuit bias. They further know, which transistors form the stage biases of the different amplification stages, the transconductors

and loads. For automation, an algorithm is needed which is able to recognize these functional blocks. However, amplification stages and their subblocks as loads, transconductors and stage biases have a high structural variety. The current state of the art, further discussed in Sec. 2.1, only provides algorithms to identify basic analog building blocks as current mirrors or differential pairs in circuits, e.g., [52]. For the recognition of amplification stages in op-amps, fixed structure libraries are used only supporting a small number of transistor structures, e.g., [53]. Thus, a new algorithm for structural recognition is needed supporting a high structural variety and allowing a fast integration of new structures for recognition.

The equation-based description of an operational amplifier is highly topology dependent. Therefore, most automation approaches on equation-based sizing are also topology dependent, e.g., [7, 54], further discussed in Sec. 2.2. For every supported topology, the analytical performance equations are stored in a library. Only a small number of topologies is supported. The influence of transistor structures on the performance equations is not considered. In Fig. 1.2a for examples, the load influences the denominator of the open-loop gain equation. Changing the load to another current mirror would change the denominator of the equation. Only a small part of the equation is affected. Nevertheless, in topology dependent approaches, as [7, 54], the Miller op-amp with the different current mirror as load would be regarded as the new topology. Its corresponding performance model would have to be set up and stored in the library. To overcome the topology dependency in equation-based sizing approaches, symbolic analysis methods, e.g., [11,13] and simulation-based sizing methods [10, 17, 20] were developed leading away from the analytical manual design process. An automatic sizing method which is topology independent but still uses the analytical equations of the manual design process for sizing was not yet developed. This method should not regard the design equations as a whole. Instead, the designer knowledge of the influence of a transistor structure on the performance equation should be integrated in the automatic equation set-up leading to a topology independent equation instantiation.

To automate the solving of the equation system, any type of solver suitable for Mixed-Integer Non-Linear Programming problems (MINLP) can be used. The sizing problem of analog circuits is a mixed-integer problem as the transistor dimensions are discrete values, the transistors current and circuit performance features are floating values as they depend non-linear on the transistor dimensions. Hence, a solver must be developed integrating these aspects.

1.2. Contributions of this Work

This work presents new approaches on analog circuit sizing and structural synthesis. Both methods are exemplarily shown on analog operational amplifiers. Their foundation lies in a complete functional analysis of the transistors in the circuit. The contributions compared to the state of the art (Chapter 2) are:

1. **A comprehensive functional block decomposition of op-amps** (Chapter 3): The decomposition is able to recognize functional blocks with a large structural variety as, e.g., the amplification stages and their subblocks as transconductors, loads and stage biases, in op-amp topologies. For each functional block, a functional and structural description is given. Algorithms for recognition were developed, which use the formalized structural description. New compared to the state of the art, thousands of op-amp

topologies can be thoroughly analyzed with this method. The functional blocks of interest to the manual analog design process are identified.

2. **An automatic initial sizing method emulating the manual design process** (Chapter 4): The functional block description of op-amps in Chapter 3 is used for a Hierarchical Performance Equation Library (HPEL). The analytical performance equations of the manual design process are not stored depending on a specific topology but are stored in a general manner. Every functional block, e.g., amplification stage, load, transconductor, is assigned a unique behavior model based on analytical equations. For a given op-amp topology, the equations are automatically specified. The HPEL overcomes the topology dependence of the state-of-the-art analytical sizing approaches. The performance model of more than 4000 topologies can be automatically instantiated. The obtained circuit model is solved using Constraint Programming. Constraint Programming is especially suitable for the combinatorial character of the analog sizing problem due to the manufacturing-induced discrete value range of transistor geometries. For the constraint programming solver, a problem specific branching heuristic was developed to make it a reliable solver for the MINLP problem of analog circuit sizing.
3. **A structural synthesis method building up topologies based on functional blocks** (Chapter 5): Based on the functional block description in Chapter 3, op-amp topologies are created based on a hierarchical composition graph. A generic algorithm is presented able to create the structural implementations of a functional block based on the structure implementation of other functional blocks. Also, an algorithm is shown to synthesize an individual bias circuit for every op-amp topology. The topologies are sized and evaluated using an enhanced version of the sizing algorithm in Chapter 4. For a given specification set, the topologies able to fulfill the performance requirements are found out of thousands of different op-amp topologies. Building op-amps based on functional blocks omits impractical and redundant topologies in the synthesis process. The topology independent sizing method allows the inclusion of more than 3900 topologies in the process.

Parts of the methods and ideas of this thesis have been pre-published. The functional block decomposition method from Chapter 3 was published in [55]. A first overview of the initial sizing method for analog operational amplifiers (Chapter 4) was given in [56]. *Abel et al., 2019* [57] gives a first idea of the automatic equation set-up of the initial sizing problem and presents a customized constraint programming solver for it. *Abel et al., 2021* [58] presents a more detailed description of the initial sizing problem and presents a problem specific branching heuristic (Sec. 4.5) developed to size more than thousand topologies with the constraint programming solver. The hierarchical performance equation library (Sec. 4.2) was published in [59]. A method to support weak, moderate and strong inversion in the initial sizing process was published in [60]. A first overview of the structural synthesis method was presented in [61]. The here presented method (Chapter 5) was published in [62]. The runtime of the algorithm was sped up using multi-threading. The corresponding method and results (Secs. 5.5 and 5.6.2) were published in [63].

1.3. Structure of the Thesis

The thesis is structured as follows. The state of the art is discussed in Chapter 2. This includes an overview of the state-of-the-art methods for the structural analysis of analog circuit (Sec. 2.1), an overview of state-of-the-art sizing methods (Sec. 2.2), and an overview of different structural synthesis algorithms (Sec. 2.3).

The functional block decomposition method is presented in Chapter 3. This includes a functional and structural description of every functional block in an op-amp (Sec. 3.1) as well as the presentation of the recognition algorithm for these functional blocks (Sec. 3.2).

The sizing method based on functional block behavior models is presented in Chapter 4. This includes a presentation of the Hierarchical Performance Equation Library (Sec. 4.2) and of the problem-specific branching heuristic for the constraint programming solver of the sizing method (Sec. 4.5).

Chapter 5 presents the structural synthesis method for analog operational amplifiers. Algorithms to synthesize the structural implementations of a functional block based on the structural implementations of other functional blocks (Sec. 5.1) as well as to synthesize the op-amp bias (Sec. 5.2) are presented. Sec. 5.4 presents the functional block composition graph and the resulting overall synthesis algorithm. The multi-threading strategies that are used to speed up the synthesis algorithm are presented in Sec. 5.5.

Experimental results for each of the three methods, functional block decomposition (Chapter 3), sizing (Chapter 4) and structural synthesis (Chapter 5), are presented at the end of the corresponding chapters. As this thesis focuses mainly on basic op-amp structures, an outlook is given how the method can be adopted to modern multi-stage op-amps (Chapter 6). The thesis ends with a conclusion (Chapter 7).

2. State of the Art

This chapter gives an overview of the state of the art of the three areas the thesis covers: structural analysis of analog circuits (Sec. 2.1), automatic sizing of analog circuits (Sec. 2.2) and structural synthesis of analog circuits (Sec. 2.3).

2.1. Structural Analysis of Analog Circuits

Structural analysis also referred to as building block analysis is the problem of finding subcircuits with a fixed topology in a circuit. Mathematically described, this is the problem of finding subgraph isomorphism. As there exist only a small number of analog building blocks with a small amount of transistors, special methods for structural analysis were developed. They are usually library-based, i.e., building blocks stored in a library are compared to a given netlist by a recognition algorithm. A well known contribution for building block analysis is [52], which is in the following section described in detail.

2.1.1. The Sizing Rules Method

The sizing rules method [52] defines a library of analog building blocks (Fig. 2.1) which is hierarchically structured. On the lowest hierarchy level L_0 are basic transistor structures as diode and normal transistors. The next three hierarchy levels build up transistor pairs based on the building blocks of the lower hierarchy levels. The building block library contains mainly current mirrors. On the highest hierarchy level (L_3) lays the differential stage.

The corresponding structural analysis algorithm (Fig. 2.2) recognizes the building blocks part of the library L in a topology. For recognition, characteristic connections of the building blocks are used. The algorithm starts with the recognition of building blocks of the lowest hierarchical level as normal and diode transistors. Then, it searches for the building blocks of the first hierarchy level. For the identification, every pair of subbuilding blocks is considered, checking if it has the characteristic connection. Thus, to identify a simple current mirror in the circuit, for every combination of normal and diode transistor, it is checked if the source pins of the transistors are connected to each other. Also, the gate of the normal transistor must be connected to the drain of the diode transistor. After searching for all structures in the circuit fulfilling the criteria of the pairs of the first hierarchy levels, it is searched for the pairs of the next higher hierarchy level. The algorithm ends by searching for differential stages in the topology.

During the analysis process, conflicts may appear. Transistors may be assigned to two different building blocks of the same hierarchy level standing in conflict to each other. For this, a dominance relation was established, giving a strict order of building blocks dominating each other. A simple current mirror for example dominates a differential pair. Thus, if a transistor is part of a simple current mirror and a differential pair, the differential pair is removed from the set of recognized structures.

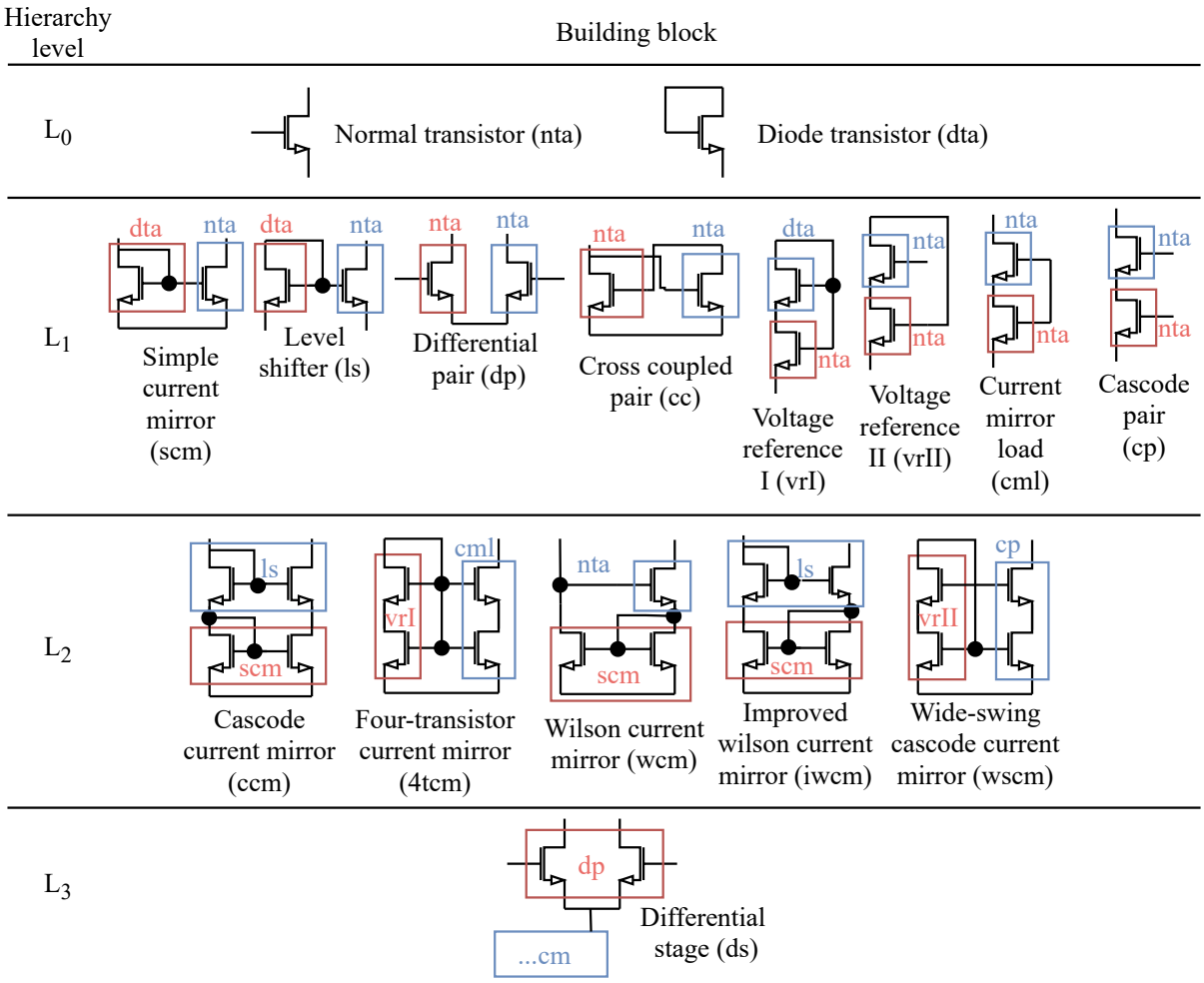


Figure 2.1.: Building block library L [52]

The algorithm ends by removing uncertain building blocks from the set of recognized building blocks. Uncertain building blocks are building blocks, which are only valid if they are part of another building block. Such a building block is for example the cascode pair. It is only valid if it is part of a wide-swing cascode current mirror.

The structural analysis of [52] was many times enhanced and altered, e.g., [64,65]. However, the building block library stayed nearly identical with the highest recognizable structure a simple differential stage. More advanced differential stages as a folded-cascode differential stage are not covered. Also building blocks with many structural representatives as amplification stages cannot be easily recognized with [52]. In some cases, it would also be useful, e.g., for constraint generation, to identify a subcircuit of an analog circuit class in a greater

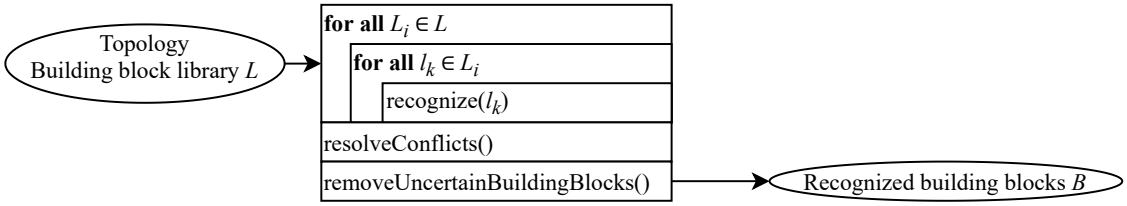


Figure 2.2.: Structure analysis according to [52]

Table 2.1.: Different building block libraries

Work	Type of library	Use-case
Massier [52]	Basic building blocks	Structural analysis, sizing
Feats [45]	Basic building blocks	Structural synthesis
Zhao [43]	Basic building blocks	Structural synthesis
GPCAD [54]	Op-amp topologies	Sizing
Maulik [36]	Op-amp topologies	Structural synthesis
Gerlach [38]	Op-amp topologies	Structural synthesis
Graupner [66]	Predefined modules, e.g., amplification stages, circuit bias	Structural synthesis, layout design
PAD [53]	Predefined modules, e.g., amplification stages, circuit bias	Sizing
Das [44]	Devices w/ or w/o self-connections	Structural synthesis
MOJITO [67]	Devices w/ or w/o self-connections	Structural synthesis

netlist, e.g., identifying an op-amp in a greater netlist. This cannot be obtained by [52].

2.1.2. Other Building Block Libraries and Structural Analysis Methods

Table 2.1 shows additional building block libraries. The libraries in [43, 45] are oriented on the building block library of [52]. Other libraries [26, 36, 38] consist of whole topologies of basic analog circuits, or predefined modules of, e.g., amplification stages or bias circuits having a certain transistor structure [46, 66]. Other libraries [44, 67] contain only single devices with or without additional self-connections.

The building block libraries in [43, 45] were developed for structural synthesis of analog circuits. This is also the case with the topology libraries in [36,38]. Other topology libraries were developed for automatic circuit sizing [26]. Predefined modules are part of analog circuit generators for structural synthesis and layout design [66] or part of tools to support the analog sizing process by design recipes [53]. Libraries containing only single devices with or without self-connections are part of the structural synthesis tools in [44, 67].

To cover complex analog structures with a high number of transistors, libraries with fixed structures have been developed, e.g., [53], Fig. 2.3, covering a small number of topologies [26, 36, 38, 53, 66]. Adding topologies or transistor structures to these libraries comes with a high set-up effort. This contrasts the high variety of op-amp topologies with thousands of variants of which only a small fraction is covered by the libraries. The library of [53] shown in Fig. 2.3, for example, supports only two types of current mirrors, a cascode and a simple current mirror. The sizing rules method in [52] in contrast supports six different current mirrors due to the hierarchical structure of the method (Fig. 2.1). However, the hierarchical library of [52] does not support greater transistor structures as the cascode differential pair and the folded-cascode differential pair supported by [53] (Fig. 2.3).

Libraries containing only single devices with or without self-connections [44, 67] come with the same problems as the building block libraries in [43, 45, 52]. They do not cover building blocks with a greater number of transistors as, e.g., amplification stages. Additional methods are needed to generate all suitable constraints for circuit sizing [20] or layout generation [28], or to omit redundant or impractical topologies in the structural synthesis process [43–45].

To overcome these issues and to include more building blocks for op-amp design in basic libraries, attempts were made using unsupervised learning algorithms [68]. However, this method is still fragile and may not recognize all building blocks correctly. Another method tries to extend the analog building blocks library by using a deterministic learning algorithm [69]. However, as the structural variety of building blocks such as amplification stages is quite

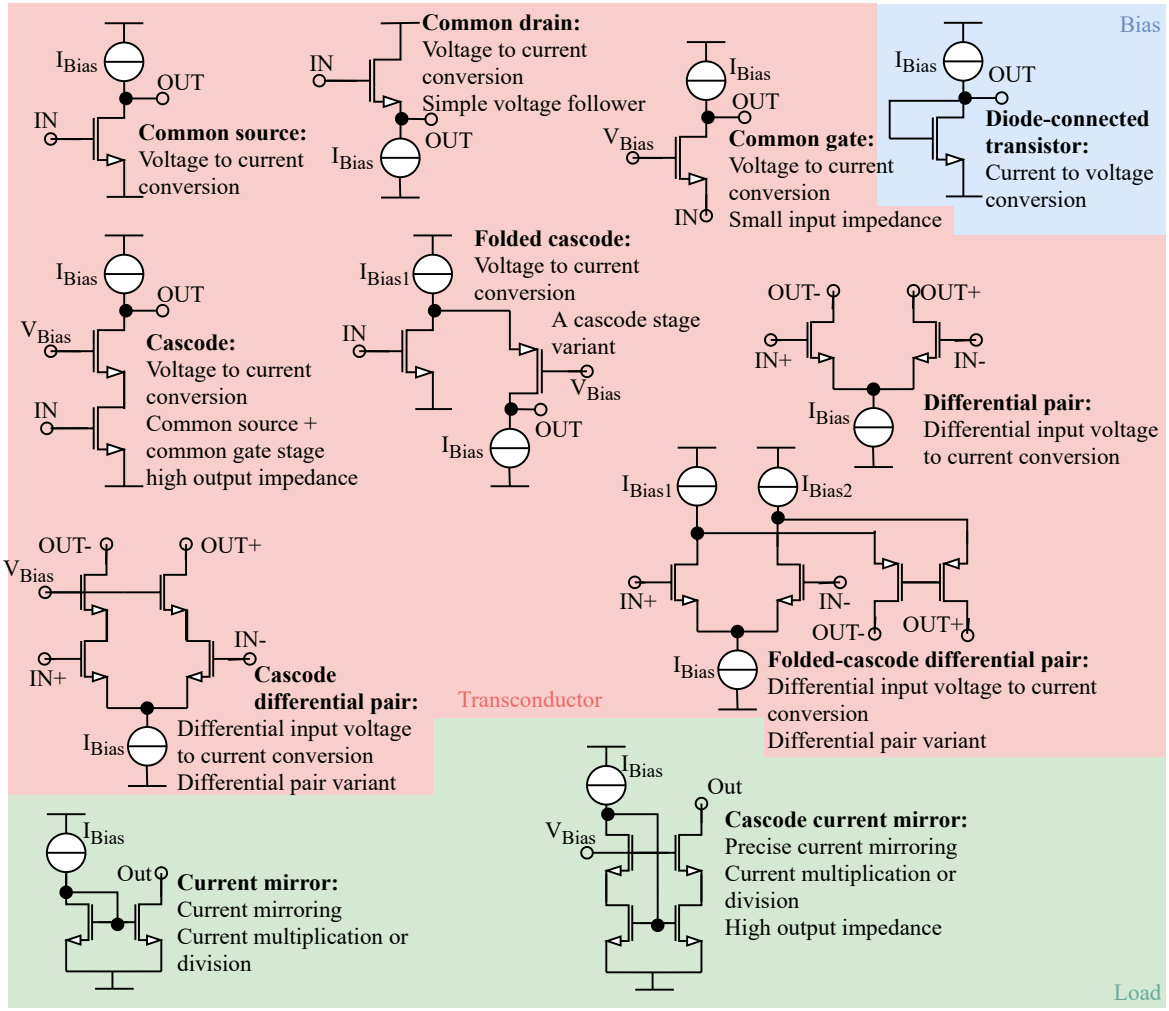


Figure 2.3.: Example of a module library [53]

high, many iterations would be needed to have a complete library covering all variations of amplification stages.

Thus, the problem of including analog building blocks with a high structural variety such as amplification stages or bias circuits in a structural recognition algorithm is still unsolved. The structure of an op-amp topology cannot yet be completely computer-aided analyzed.

2.2. Sizing Methods for Analog Circuits

Sizing of analog circuits yields to determine the circuit's design parameters, e.g., width and length of a transistors, so that the performance requirements, that must be kept, e.g., for minimum gain, maximum power consumption, are fulfilled. The industrial practice of analog sizing is manual and equation-based, followed by simulation, corner-case simulation and Monte-Carlo analysis for verification. To automate the sizing of analog circuits, different methods were developed. Two main groups can be distinguished: equation-based sizing and simulation-based sizing.

Table 2.2.: Overview of state-of-the-art equation-based sizing approaches

Tool/Author	Supported circuits	Equation set-up	Set-up time	Execution time
IDAC [7]	Amplifiers	Topology database	Months	Few seconds
BLADES [8]	Op-amps, subcircuits	Subcircuit-based	Long	20 min
OPASYN [9]	Op-Amps	Topology database	Few weeks	5 min
Maulik [12]	Op-Amps	Topology database	Long	Few seconds
GPCAD [54]	Op-amps	Topology database	Long	Few seconds
Leyn [10]	Amplifiers	Symbolic Analysis	Hours	Few seconds
AMGIE [11]	Op-Amps	Topology database, Symbolic analysis	8 h	20 min
Shi [13]	Op-Amps	Symbolic analysis	Few seconds	-
Verhagen [14]	Op-Amps	Symbolic analysis	Few seconds	-
Liu [15]	Op-Amps	Symbolic analysis	Few seconds	Few seconds
Binkley [70]	Transistors	Fixed implemented transistor equations	-	-
PAD [53]	Op-Amps	Subcircuit-based	Long	-

2.2.1. Equation-Based Sizing Methods

Many computer-aided approaches for sizing are equation-based, e.g., [7–15, 54, 57, 58]. The ac-behavior, dc-behavior, and transient behavior of the circuit are modeled by analytical equations using similar equations as in the analog manual design process. The methods aim at finding a first, i.e., initial sizing of the circuit considering the performance specifications.

The major obstacle with design equations is the effort to set them up. Equation-based sizing approaches (Table 2.2) have presented fixed design plans for supported process technologies [7–9, 12, 54], or apply symbolic analysis to create transfer functions automatically [10, 11, 13–15]. The methods support mainly op-amps [8, 9, 11–15, 54, 57, 58] and other types of amplifiers [7, 10].

Early equation-based methods [7, 9, 12, 54] stored the equations topology dependent using the same equations to model the ac-behavior, dc-behavior and transient behavior as in the manual design process. Topology libraries were developed containing a fixed equation set for every supported topology. The design plans contain the analytical equations describing the behavior of the corresponding circuit as well as additional constraints needed for sizing, e.g., dimension constraints. Solving the design plans for a given set of specifications takes in many cases only seconds.

In [54], for example, an automatic equation-based sizing method supporting five different op-amp topologies (e.g., Fig. 1.2a) is presented. For each circuit, a design plan is given containing equations for the included constraints and specifications (Table 2.3). These are sizing as well as performance constraints including analytical equations to model the ac-behavior, dc-behavior and transient behavior of the circuit. The models implemented in the fixed design plan are solved with geometric programming to find the component dimensions of the circuits. The component sizes are optimized to maximize chosen performance safety margins. As geometric programming only supports posynomial equations, the analytical equations must be adopted to fulfill that form. In some cases, e.g., for the phases margin, approximation must be made to obtain a posynomial equation. The equations are highly topology related. No comments are made, how the equation changes if another load or second stage is used in a two-stage op-amp. As transistor model, a simplified version of the Shichman-Hodges model [71] is used. It is a posynomial model and thus does not change the character of the equations.

The early equation-based sizing methods have the disadvantages that the programming of a circuit-specific design plan must be done for each individual circuit topology and takes a

Table 2.3.: Design constraints and specifications in [26]

Specification/Constraint		Type
Transistor dimensions	Symmetry and matching	Monomial
	Device sizes	Monomial
	Area	Posynomial
	Systematic offset voltage	Monominal
Performance requirements	**dc-behavior**	
	Common-mode input voltage range	Posynominal
	Output voltage swing	Posynomial
	Quiescent power	Posynomial
	ac-behavior	
	Open-loop gain	Monomial
	Dominant pole conditions	Posynomial
	Unity-gain bandwidth	Monomial
	Phase margin	Posynomial
	CMRR	Monomial
	Neg. PSRR	Inverse posynomial
	Pos. PSRR	Neither
	Input-referred spot noise	Posynomial
	Input-referred total noise	Posynomial
	Transient behavior	
	Slew rate	Posynomial

long time. To overcome the topology dependence, *El-Turky and Perry* [8] split up part of the equation-based description into subcircuits descriptions. However, only basic equations, as symmetry constraints and dc-performance constraints, are considered for the subcircuits. ac-performance and transient performance constraints are still restricted to a specific op-amp topology. Adding new topologies to these methods takes quite long as for every topology a new equation-based description has to be developed.

To reduce the set-up time of the equation-based description, symbolic analysis methods were developed [10, 11, 13–15]. They automatically create the transfer function of a given topology. The transfer function, however, only represents the ac-behavior of the circuit. To represent the transient behavior and dc-behavior of the circuit, other methods are still needed. Designer knowledge is for instance used in [11] to include performance features for the transient behavior and symmetry constraints in the equation-based model. This results in a topology library with reduced set-up time for new topologies.

Other approaches to automate the equation-based sizing process provide graphical support for an interactive design through the design equations [53, 70]. *Binkley et al.* [70] deals with the behavior of a single transistor, describing its behavior by analytical equations. *Stefanovic and Kayal* [53] present a method to guide through the manual analog design process by using design recipes for well defined analog structures (Fig. 2.3). To simplify the analog design problem, a circuit is partitioned into modules, i.e., subcircuit of predefined topologies. The modules are part of a library (Fig. 2.3). Every module has a defined topology as well as functionality. The tool provides insights into the influence of the different modules on the circuit behavior, i.e., it presents the design parameters of each module and how the module must be designed to fulfill a certain function in the circuit. For a fixed number of op-amp topologies, the analytical equations of the performance features as, e.g., open-loop gain, phase margin, are implemented in design plans. As transistor model, the EKV model is used. The designer can separately design the modules until all specifications are fulfilled.

The main drawback of the state-of-the-art approaches is the effort to set them up. As many tools have fixed design plans for each topology, the behavioral equations must be set up from scratch for every new topology. This takes long and is a major disadvantage as the

equation set-up is not automated. An automatic set-up of all necessary design equations of the manual design process has not been published.

2.2.2. Simulation-Based Sizing Methods

Simulation-based sizing methods [17, 19, 20, 22–24, 45, 72, 73] optimize the components dimensions of a circuit towards fulfilling the circuit specifications, performance safety margins and/or a higher yield using numerical methods. As simulations are used to evaluate the circuit behavior, the methods are topology independent. Also, advanced transistor models are used in the simulations such that a higher accuracy in the behavior circuit models is obtained. The approaches [72] and [17] have been transferred into commercial design suites of the companies MunEDA [74] and Cadence [5], respectively. These methods also include advanced design aspects like manufacturing variation and yield optimization, as well as aging or Pareto optimization.

Simulation-based sizing needs an initial sizing that provides some reasonable performance values to start from. Most optimization methods today use numerical methods to find initial sizings [72] or use random generated values as start dimensions [73].

Although simulation-based sizing provides topology independence, potentially better accuracy and results, designers prefer the traditional way of initial, equation-based sizing. There may be two reasons for this: One is the lack of physics orientation and lack of insight that comes with numerical methods. The other is the effort to set up the numerical sizing tool. All functional and dimension constraints must be set up for every topology before the start of the sizing procedure as they are needed to guide the numerical optimization process to meaningful solutions.

2.3. Structural Synthesis of Analog Circuits

Structural synthesis of analog circuits describes the process of finding a circuit topology and its corresponding set of parameters, e.g., transistor widths and lengths, such that a given set of specifications is fulfilled. Two paths can be distinguished: One path of structural synthesis is a specification-based selection of one or few netlists from a library [36–39, 41, 42]. This is an automation of what happens in practice. Every company has netlist sets for the different analog circuit types, e.g., 20 to 30 different op-amp topologies. According to given specifications, it is chosen from the set. An experienced designer can do this selection instantly such that there is little gain in design time or design quality through the automation of this process. The other path of structural synthesis builds up the netlist by combining modules of transistors and transistor groups while satisfying Kirchhoff's laws and basic voltage/current conversion at the module interfaces [35, 40, 43–46]. Here, a plethora of variants is created, from which the promising ones are selected only after symbolic analysis and complete sizing. Designers see an unnecessary variety of intermediate solutions that they would never have created.

Early developed structural synthesis approaches were [36, 39, 46]. *Maulik et al.* [36] and *Harjani et al.* [39] developed structural synthesis tools for operational amplifiers. In [36], the method chooses from a library of 64 topologies using fixed design plans for evaluation. *Harjani et al.* [39] present a top level abstraction of a circuit which is hierarchically transformed

Table 2.4.: Comparison of state-of-the-art structural synthesis algorithms for op-amp design

Work	# supported topologies	Search duration	Type of sizing
Maulik [36]	64	< 1 h	Equation-based
OASYS [39]	~70	< 0.2 h	Equation-based
Gerlach [38]	24	< 0.5 h	Numerical
Koza [35]	> 10 000	2 d	Numerical
MOJITO-R [40]	~3500	15 h	Numerical
Das [44]	> 10 000	N.a.	Numerical
Feats [45]	> 10 000	8 h	Numerical
Zhao [43]	~4000	Secs. (w/o sizing)	Numerical

into a transistor level structure. As in [36], fixed design plans based on analytical equations are used for evaluation. The approach in [46] generates LC-structures or switch-capacitor filters using symbolic analysis for evaluation. To overcome the topology dependence in [36,39], synthesis approaches based on genetic algorithms were developed [35, 40, 41, 75]. Instead of libraries of whole topologies, libraries containing simple transistor structures are used. Based on these building block libraries, many different circuit topologies are created using genetic programming. However, many topologies are redundant or impractical. Simulations are used to evaluate the created topologies and to remove the impratical ones. Graph-grammar-based approaches using strict grammar rules and isomorphism techniques were developed [43–45] to lessen the number of redundant topologies. The evaluation and sizing of the topologies takes place after synthesizing all possible topologies. Thus, much computation time is spent for circuits which cannot fulfill the given specifications from a visual inspection of a designer. Other approaches to lessen the number of created topologies are rule-based topology synthesis algorithms which closely implement designer knowledge [37, 38, 42].

As this work focuses on the structural synthesis of operational amplifiers, this paragraph takes a closer look on synthesis tools supporting operational amplifiers. A selection of synthesis tools for op-amp design are listed in Table 2.4. The table presents the number of supported topologies of each synthesis tool, the runtime of the algorithms and the used type of sizing. State-of-the-art synthesis approaches which have only a small number of fixed topologies implemented [36, 38, 39] are fast in their evaluation. As design knowledge is used in the evaluation method, less than one hour is needed to find circuits for a given set of specifications. The synthesis tools creating a plethora of circuits [35, 40, 45] have high evaluation times of more than a work day. For many tools, the search duration was obtained using a reduced building block library such that fewer topologies than the maximum possible number were created. Also, advanced hardware and parallelization techniques were used to reduce the runtime. Thus, a structural synthesis method supporting many topologies but still fast in evaluation is still missing.

3. Functional Block Decomposition[1]

The chapter presents a method to recognize functional blocks in op-amps. Different to [52], it does not stop with the recognition of the differential stages (Sec. 2.1). Instead, it is able to identify whole amplification stages with their transconductors, loads and stage biases in an op-amp. It also recognizes all transistors forming the bias circuit of an op-amp. Hence, it is able to completely analyze an op-amp topology. The method does not use structural pairs as [52], but builds up on a unique structural description for every functional block in an op-amp. With the descriptions, many structural representations of a functional block are covered. As in [52], the analysis is hierarchical. For the basic analog building blocks on the lower hierarchy level, similar to [52] dominance relations are needed to remove conflicts and uncertain building blocks.

In the following section, the functional building blocks in an op-amp are described (Sec. 3.1). For each functional block, a formalized functional and structural description is presented. The structural description of the functional blocks is used in the functional block analysis to recognize the functional blocks automatically (Sec. 3.2). Results obtained by the functional block analysis on four different circuits are shown in Sec. 3.3.

3.1. Functional Blocks in Op-Amps

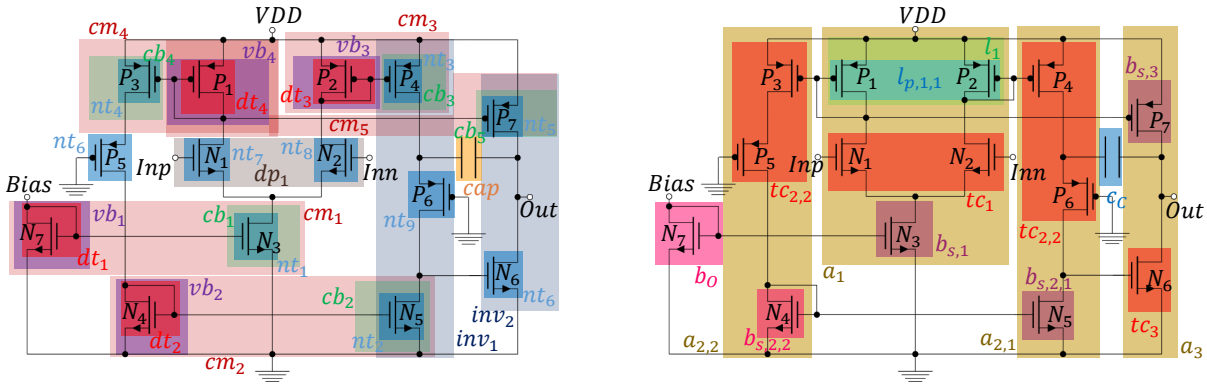
An op-amp can be hierarchically decomposed into functional blocks. Fig. 3.1 shows this decomposition. Starting with one functional block at the highest hierarchy level (HL 5), the number of functional blocks per level increases till on the lowest level (HL 1) every device of the circuit forms a functional block of its own. On HL 1, every functional block is represented by one uniquely definable device composition. However, this level does not give any information about which device takes up which functional task in an op-amp, e.g. amplification, stabilization, biasing. This functional assignment is given at HL 3 and HL 4. At these levels, the function of every functional block is uniquely definable. However, the structural description is not unique as, e.g., different types of amplification stages exist. On HL 2, neither the structural nor the functional definition of the functional blocks is unique. However, their structural complexity is less then those of the functional blocks of HL 3 and HL 4.

Fig. 3.2 shows as an example the functional blocks in a symmetrical op-amp. For HL 1 and HL 2 (Fig. 3.2a), all functional blocks of the same type have similar device compositions. This is different for HL 3 and HL 4 (Fig. 3.2b). The composition of the first stage a_1 differs highly from the composition of the other stages. On HL 1, HL 3, and HL 4, all devices can only be part of one functional block on a level. N_5 is a normal transistor nt_2 on HL 1, a stage bias $b_{2,1}$ on level 3 and part of an amplification stage $a_{2,1}$ on level 4. On HL 2, a device can

[1]The chapter was similarly published in [55], reprinted with permission from "I. Abel, M. Neuner, and H. Graeb, A Functional Block Decomposition Method for Automatic Op-Amp Design, Integration" © 2022 Elsevier.

HL 5	Op-amp	Functional abstraction ↑
HL 4	Amplification stage (a), circuit bias (b_O), compensation capacitor (c_C), load capacitor (c_L)	
HL 3	Transconductor (tc), load (l), stage bias (b_S),	
HL 2	Voltage bias (vb), current bias (cb), current mirror (cm), differential pair (dp), analog inverter (inv)	Structural refinement ↓
HL 1	Normal transistor (nt), diode transistor (dt), capacitor (cap),	

Figure 3.1.: Hierarchical library of functional blocks in op-amps



(a) Functional blocks on level 1 - 2

(b) Functional blocks on level 3 - 4

Figure 3.2.: Symmetrical op-amp with high Power-Supply Rejection Ratio (PSRR) [48] (Definitions of abbreviations in Fig. 3.1)

be part of more than one functional block. On this level, N_5 is at the same time a current bias cb_2, part of a current mirror cm_2 and part of an analog inverter inv_1.

We define the set of functional blocks $\mathcal{X}$ in an op-amp as:

$$\mathcal{X} = \{x_k | k = 1, 2, .., |\mathcal{X}|\} \tag{3.1}$$

A functional block x_k in a circuit consists either of a basic device d_k of type $d_k.type \in \{t, c\}$, where t is referring to transistors and c to capacitors, or of other functional blocks $x_{k,1}, .., x_{k,n}$ of individual types $x_{k,l}.type \in \{dt, nt, cap, vb, cb, cm, dp, inv, g, l, b, a, c_L, c_C\}$ (see Fig. 3.1):

$$\forall_{x_k \in \mathcal{X}} \ (x_k = \{d_k\} \vee x_k = \{x_{k,1}, .., x_{k,n}\}) \tag{3.2}$$

The subset $\mathcal{X}_j \subseteq \mathcal{X}$ contains all functional blocks $x_k \in \mathcal{X}$ with $x_k.type = j$.

To describe the connections between functional blocks, every functional block x_k is assigned a set of pins P_{x_k}, which are connected to the nets of the circuit:

$$P_{x_k} = \{x_k.p_l | l = 1, 2, ..|P_{x_k}|\} \tag{3.3}$$

As all functional blocks consist at the end of devices, the pins P_{x_k} of a functional block x_k refer to certain device pins.

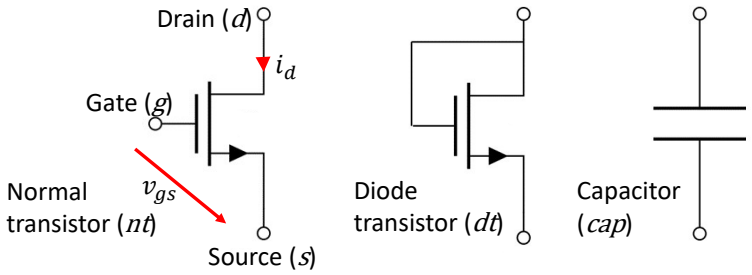


Figure 3.3.: Functional blocks on HL 1

A connection of two functional blocks x_k, x_l over any net with the pins $x_k.p_y, x_l.p_z$ is described by:

$$x_k.p_y \leftrightarrow x_l.p_z \tag{3.4}$$

To describe that two pins $x_k.p_y, x_l.p_z$ are not allowed to be connected by any net, the following notation is used:

$$x_k.p_y \nleftrightarrow x_l.p_z \tag{3.5}$$

$x_k.\Phi$ denotes the substrate type of a functional block x_k, with the following naming convention:

$$x_k.\Phi \in \{\Phi_n, \Phi_p, \Phi_u\}, \quad \text{for n-, p-, or mixed-doping} \tag{3.6}$$

A functional block x_k has mixed-doping if it consists of transistors with different doping.

In the following sections, structural and functional definitions will be given for all functional block types in Fig. 3.1. Please note that all examples shown for NMOS transistors hold analogously for PMOS transistors.

3.1.1. Functional Blocks on Hierarchy Level 1

Fig. 3.3 shows three different device level functional blocks. In addition to capacitors, we define and use two different functional block types for transistors:

Normal Transistor

Function: A normal transistor nt_k establishes a relation between the voltage potential v_g at the gate of the transistor and the current at the drain i_d of the transistor. Either v_g or i_d is the control factor.

Structure: A normal transistor nt_k is a transistor d_k without any self-connections.

$$x_k = \{d_k\} \wedge d_k.type = t \wedge d_k.d \nleftrightarrow d_k.s \wedge d_k.d \nleftrightarrow d_k.g \wedge d_k.g \nleftrightarrow d_k.s\} \Leftrightarrow x_k.type = nt \tag{3.7}$$

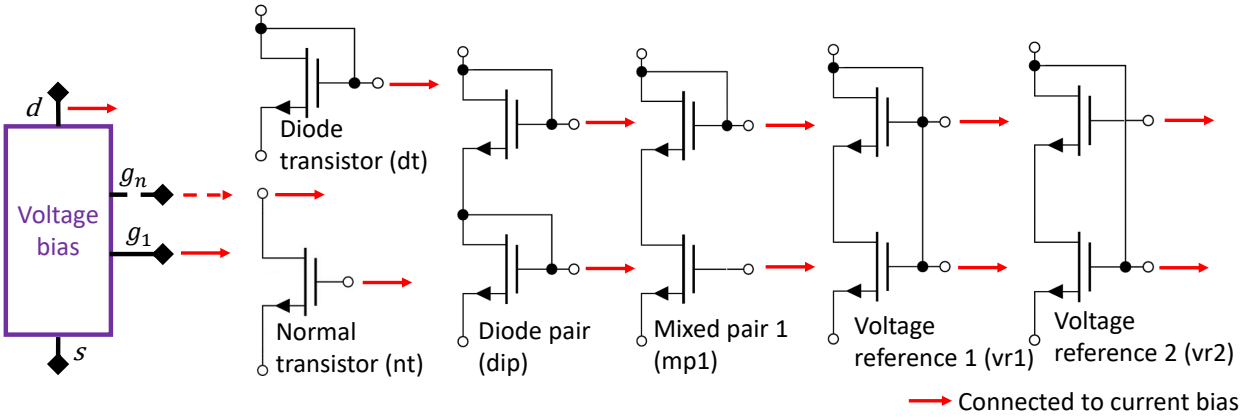


Figure 3.4.: Voltage bias and variants with stacks of 1 or 2 transistors (dashed lines: optional pins)

Diode Transistor

Function: A diode transistor dt converts its drain-source current i_d into a stable gate-source voltage v_{gs}.

Structure: A diode transistor dt_k is a transistor d_k, whose drain $d_k.d$ is connected to its gate $d_k.g$:

$$x_k = \{d_k\} \wedge d_k.type = t \wedge d_k.d \leftrightarrow d_k.g \wedge d_k.d \nleftrightarrow d_k.s \Leftrightarrow x_k.type = dt \tag{3.8}$$

3.1.2. Functional Blocks on Hierarchy Level 2

The majority of the functional blocks on HL 2 consist of transistor stacks. Typical transistor stacks are shown in Figs. 3.4, 3.5. A transistor stack ts_k is a set of 1-3 transistors having the same doping and a drain-source connection, i.e., the drain of a lower transistor in the stack $x_{k,m}.d$ is connected to the source of the next higher transistor $x_{k,m+1}.s$. Higher transistor gates are not allowed to be connected to drains of lower transistors. Drains of higher transistors are not allowed to be connected to lower transistor sources.

$$\begin{aligned} x_k = &\{x_{k,1}, .., x_{k,n} | n = |x_k| \wedge n \leq 3\} \wedge x_k \subset (\mathcal{X}_{nt} \cup \mathcal{X}_{dt}) \wedge x_{k,m}.d \leftrightarrow x_{k,m+1}.s \\ &\wedge x_{k,m}.\Phi = x_{k,m+1}.\Phi \wedge x_{k,m+1}.g \nleftrightarrow x_{k,m}.d \wedge x_{k,m+1}.d \nleftrightarrow x_{k,m}.s \Leftrightarrow x_k.type = ts \end{aligned} \tag{3.9}$$

The usual number of transistors in a stack is 1 or 2. For the pins in a transistor stack, the following naming convention will be used. The source not connected to any drain in the stack $ts_k.s_1$ is the source $ts_k.s$ of the transistor stack. The drain not connected to any source of the stack $ts_k.d_n$ is the drain $ts_k.d$ of the transistor stack. If all drains or sources of the stack must be considered, numbering will be used. The definition of the transistor stack is hereinafter used to describe the functional blocks in detail.

Voltage Bias

Function: A voltage bias vb_k converts its drain current i_d into a stable gate-source voltage v_{gs} applied to a gate of a current bias.

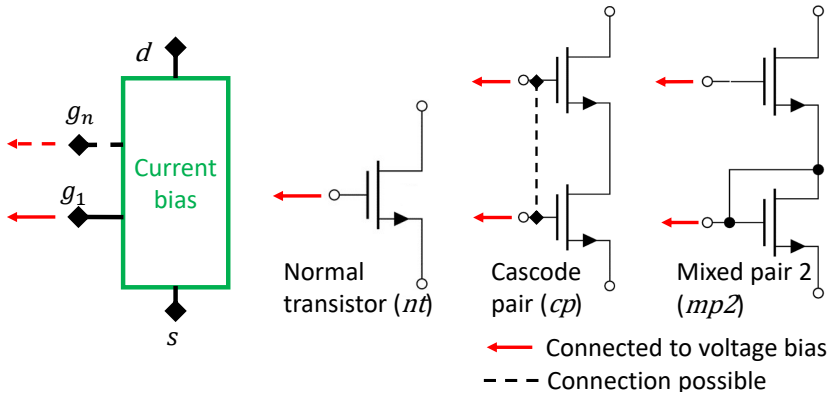


Figure 3.5.: Current bias and variants with stacks of 1 or 2 transistors (dashed lines: optional)

Structure: A voltage bias vb_k (Fig. 3.4) consists of a transistor stack $x_k, x_k.type = ts$, with the gates of its devices $x_k.g_l$ connected to gates of a current bias cb_v with same doping as vb_k. Additionally, the drain of the stack $x_k.d = x_{k,n}.d$ must be connected to a gate $cb_v.g_m$ of cb_v. For every gate $x_k.g_j$ in the stack, exactly one gate-drain connection with another transistor $x_y \in (\mathcal{X}_{nt} \cup \mathcal{X}_{dt})$ of same doping exists. This transistor x_y must be part of a voltage or current bias $x_z \in (\mathcal{X}_{vb} \cup \mathcal{X}_{cb})$ but not necessarily of vb_k or cb_v itself.

$$\begin{aligned} x_k =& \{x_{k,1}, .., x_{k,n} | n = |x_k|\} \wedge x_k.type = ts \wedge \exists_{cb_v} \big[cb_v.\Phi = x_k.\Phi \wedge x_{k,n}.d \leftrightarrow cb_v.g_m \\ & \wedge \forall_{x_k.g_l} [x_k.g_l \leftrightarrow cb_v.g_i] \big] \wedge \forall_{x_k.g_j} \big[\exists!_{x_y \in (\mathcal{X}_{nt} \cup \mathcal{X}_{dt})} [(x_y.\Phi = x_k.\Phi \wedge x_y.d \leftrightarrow x_k.g_j) \\ & \leftrightarrow \exists!_{x_z \in (\mathcal{X}_{vb} \cup \mathcal{X}_{cb})} x_y \in x_z] \big] \Leftrightarrow x_k.type = vb \end{aligned} \tag{3.10}$$

Current Bias

Function: A current bias cb_k converts the voltage potential v_g applied at its gates into a drain current i_d.

Structure: A current bias cb_k (Fig. 3.5) consists typically of a stack of normal transistors, which might have a gate connection. In rare cases, the normal transistor at the bottom is exchanged by a diode transistor. The gates of a current bias $cb_k.g_l$ are connected to a gate or the upper drain of a voltage bias $(vb_v.g_l \vee vb_v.d)$. The voltage bias must have the same doping. The drain of the upper transistor of a current bias $cb_{k,n}.d = cb_k.d$ must not be connected to any gate of a voltage bias $vb_y.g_z$ with the same doping.

$$\begin{aligned} x_k =& \{x_{k,1}, .., x_{k,n} | n = |x_k|\} \wedge x_k.type = ts \wedge \exists_{vb_v} \big[vb_v.\Phi = x_k.\Phi \\ & \wedge \forall_{x_k.g_l} [x_k.g_l \leftrightarrow (vb_v.g_l \vee vb_v.d)] \big] \wedge \nexists_{vb_y} [vb_y.\Phi = x_k.\Phi \wedge x_{k,n}.d \leftrightarrow vb_y.g_z] \\ & \Leftrightarrow x_k.type = cb \end{aligned} \tag{3.11}$$

If certain requirements are fulfilled, a voltage bias and a current bias form a current mirror.

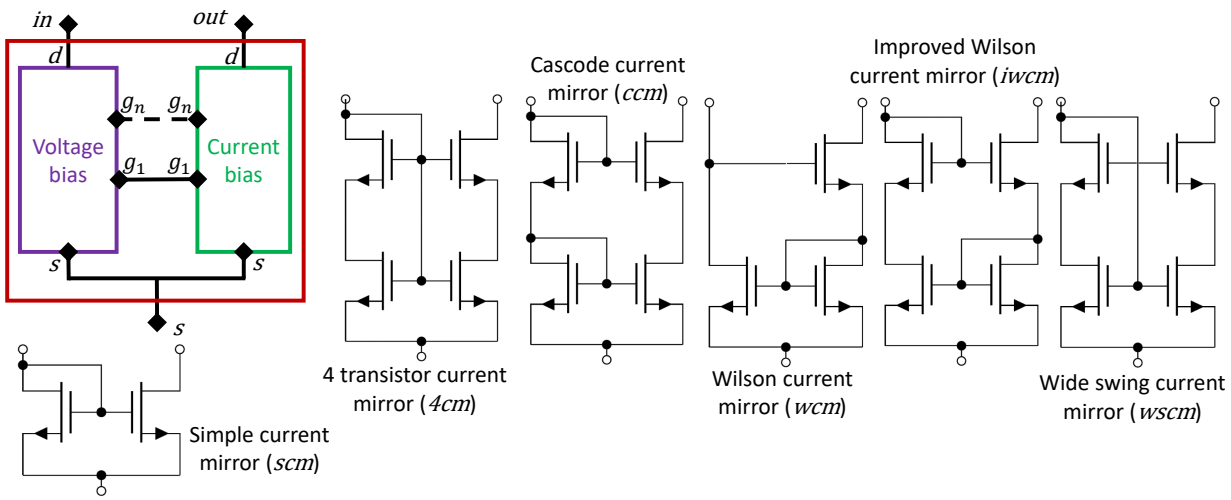


Figure 3.6.: Current mirror and examples (dashed lines: optional)

Current Mirror

Function: A current mirror cm_k provides a current by the current bias, specified by the device sizes and the current of the voltage bias.

Structure: Fig. 3.6 shows the structural definition of a current mirror and examples. To form a current mirror cm_k, voltage bias vb_k and current bias cb_k must have equal doping and a source connection. The gates of the voltage bias $vb_k.g_l$ must be connected to the gates $cb_k.g_l$ of the current bias. The uppermost drain of the voltage bias $vb_k.d$ must be connected to a gate of the current bias $cb_k.g_m$. All gates of the voltage bias $vb_k.g_j$ with exception of the upper most one $vb_k.g_{|vb_k|}$ must have a connection to a drain of either voltage or current bias $x_y.d_z|_{x_y \in \{vb_k, cb_k\}}$.

$$\begin{aligned} x_k =& \{vb_k, cb_k\} \wedge vb_k.\Phi = cb_k.\Phi \wedge vb_k.s \leftrightarrow cb_k.s \wedge (vb_k.g_l \leftrightarrow cb_k.g_l)|_{l \leq |vb_k|} \\ & \wedge \exists_{cb_k.g_m}[cb_k.g_m \leftrightarrow vb_k.d] \wedge \forall_{vb_k.g_j \in P_{vb_k} \setminus \{vb_k.g_{|vb_k|}\}} \left[\exists!_{x_y.d_z \in P_{x_y}|_{x_y \in \{vb_k, cb_k\}}} \right. \\ & \left. [vb_k.g_j \leftrightarrow x_y.d_z]\right] \Leftrightarrow x_k.type = cm \end{aligned} \tag{3.12}$$

Differential Pair

Function: A differential pair dp_k converts the voltage potentials at the gates of its two input transistors into amplified drain currents. Equal voltages lead to equal drain currents. Depending on the structure, higher amplification gains can be obtained. Cascode versions with four transistors have a higher amplification gain than a simple version with two transistors.

Structure: Fig. 3.7 shows the structure of a differential pair. The basic structure is the simple differential pair which can also stand alone.

A *simple differential pair* dp_k consists of two normal transistors $nt_{k,1}, nt_{k,2}$ connected only at their sources. This common source $dp_k.s$ must be connected to a current bias drain $cb_l.d$. The two normal transistors and the current bias must have equal doping.

$$\begin{aligned} x_k =& \{nt_{k,1}, nt_{k,2}\} \wedge nt_{k,1}.\Phi = nt_{k,2}.\Phi \wedge nt_{k,1}.s \leftrightarrow nt_{k,2}.s \wedge nt_{k,1}.d/g \nleftrightarrow nt_{k,2}.d/g \\ & \wedge \exists_{cb_l}[nt_{k,1}.s \leftrightarrow cb_l.d \wedge nt_{k,1}.\Phi = cb_l.\Phi] \Leftrightarrow x_k.type = dp \end{aligned} \tag{3.13}$$

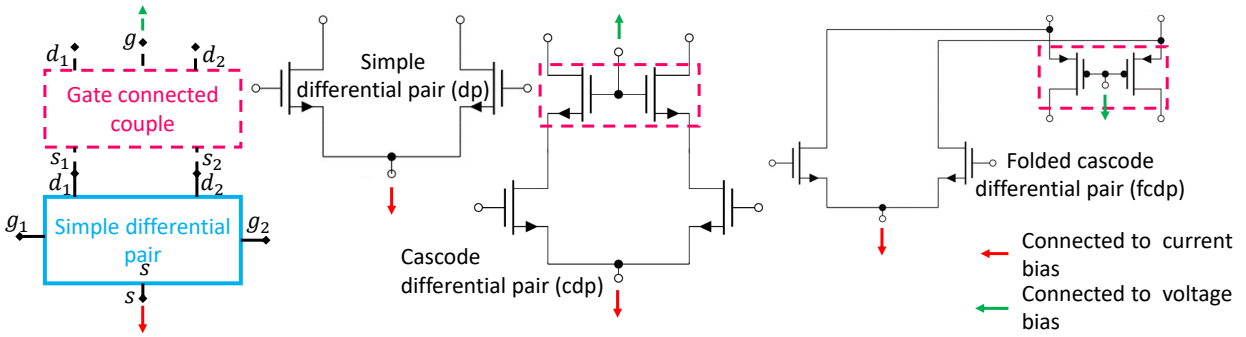


Figure 3.7.: Differential pair and examples (dashed lines: optional)

For the cascode version of the differential pair vdp_k, a simple differential pair dp_k is connected with its drains to the sources of a *gate connected couple* gcc_k. These are two normal transistors with same doping only connected at their gates:

$$\begin{aligned} x_k =& \{nt_{k,1}, nt_{k,2}\} \wedge nt_{k,1}.\Phi = nt_{k,2}.\Phi \wedge nt_{k,1}.g \leftrightarrow nt_{k,2}.g \wedge nt_{k,1}.d/s \nleftrightarrow nt_{k,2}.d/s \\ & \Leftrightarrow x_k.type = gcc \end{aligned} \tag{3.14}$$

The structural definition of the *cascode version of the differential pair* vdp_k then is:

$$x_k = \{dp_k, gcc_k\} \wedge (dp_k.d_l \leftrightarrow gcc_k.s_l)|_{l=1,2} \wedge dp_k.s/g_{1,2} \nleftrightarrow gcc_k.g/d_{1,2} \Leftrightarrow x_k.type = vdp \tag{3.15}$$

Two types of cascode variants exist having different doping characteristics. In the *folded cascode differential pair* $fcdp_k$, dp_k and ggc_k have opposite doping:

$$vdp_k = \{dp_k, gcc_k\} \wedge dp_k.\Phi \neq gcc_k.\Phi \Leftrightarrow vdp_k.type = fcdp \tag{3.16}$$

While in the *cascode differential pair* cdp_k, they have equal doping:

$$vdp_k = \{dp_k, gcc_k\} \wedge dp_k.\Phi = gcc_k.\Phi \Leftrightarrow vdp_k.type = cdp \tag{3.17}$$

Analog Inverter

Function: An analog inverter inverts and amplifies an input voltage applied at one of its gates.

Structure: An analog inverter inv_k (Fig. 3.8) consists of two normal transistor stacks $ts_{k,1}, ts_{k,2}$ differing in doping. The stacks are connected at their drains $ts_{k,1}.d, ts_{k,2}.d$. The two sources $ts_{k,1}.s, ts_{k,2}.s$ are connected to the supply voltage rail that corresponds to the doping type. Gate-gate, gate-drain and source-source connections between transistors are not allowed.

$$\begin{aligned} x_k =& \{ts_{k,1}, ts_{k,2}\} \wedge (ts_{k,1} \cup ts_{k,2}) \subset \mathcal{X}_{nt} \wedge ts_{k,1}.d \leftrightarrow ts_{k,2}.d \wedge ts_{k,1}.\Phi = \Phi_p \\ & \wedge ts_{k,2}.\Phi = \Phi_n \wedge ts_{k,1}.s \leftrightarrow VDD \wedge ts_{k,2}.s \leftrightarrow GND \\ & \wedge \forall_{nt_i, nt_j \in (ts_{k,1} \cup ts_{k,2})} [nt_i.g \nleftrightarrow nt_j.g \wedge nt_i.d \nleftrightarrow nt_j.g \wedge nt_i.s \nleftrightarrow nt_j.s] \Leftrightarrow x_k.type = inv \end{aligned} \tag{3.18}$$

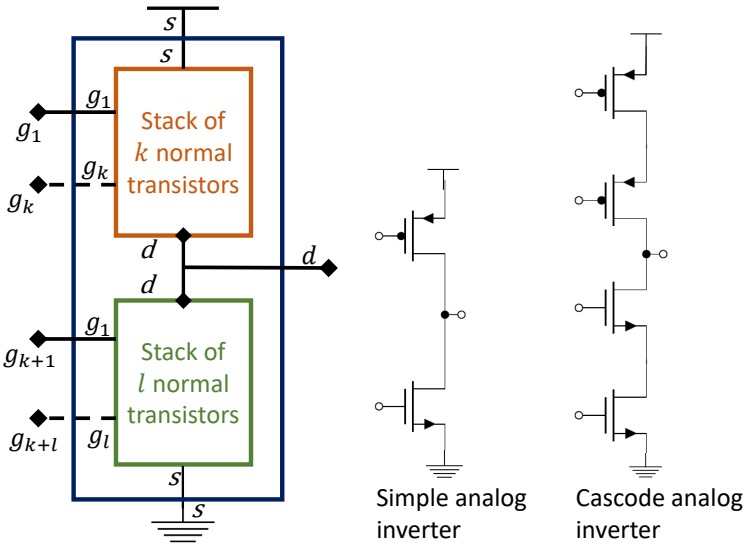
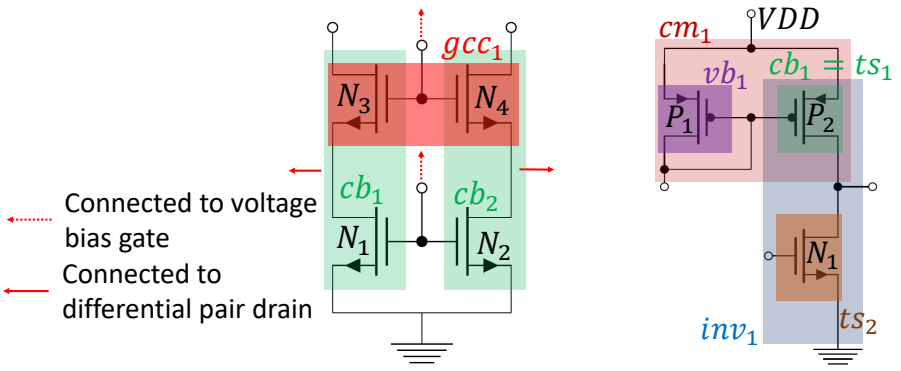


Figure 3.8.: Analog inverter and examples (dashed lines: optional)

Multiple Assignments of Transistors to Functional Blocks

Hierarchy level 2 is the only level which allows transistors to be part of more than one functional block. An example are current mirrors. According to (3.12), every transistor in a current mirror $t_k \in cm_k$ is at the same time part of a voltage or current bias $t_k \in (vb_k \cup cb_k)$. We distinguish between relevant, irrelevant and false multiple assignments.

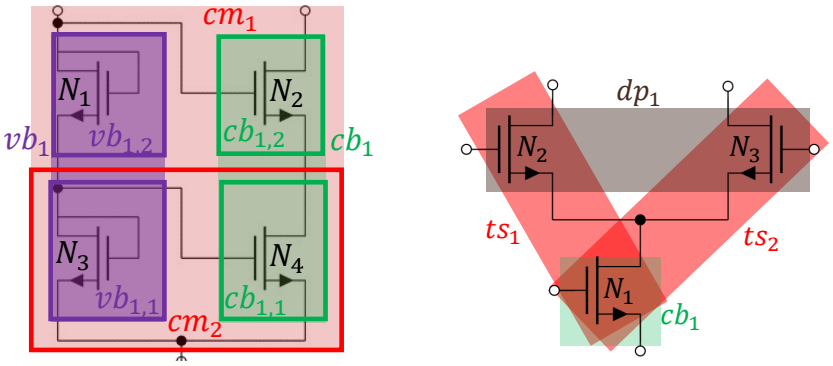
Relevant multiple assignments are circuitry-wise correct and obtain additional information needed to find other functional blocks. These are the above mentioned double assignments in current mirrors, current mirrors with the same voltage bias but different current biases forming a current mirror bench, and the two cases shown in Fig. 3.9. Fig. 3.9a shows a gate connected couple (3.14) gcc_1 in two current biases cb_1, cb_2. The gate connected couple must be part of a cascode version of a differential pair vdp_1 (3.15). The transistors N_3, N_4 are therefore part of gcc_1, vdp_1 and cb_1 or cb_2. Fig. 3.9b shows an analog inverter with a transistor stack $ts_1 = \{P_2\}$ that is also part of a current mirror cm_1. Thus, P_2 is part of inv_1, cm_1 and the current bias cb_1 in cm_1.



(a) Current biases with gate connected couple

(b) Inverter with current mirror

Figure 3.9.: Relevant multiple assignments



(a) Simple current mirror in a cascode version

(b) Cascode pairs in a differential pair with current bias

Figure 3.10.: Irrelevant (a) and false (b) multiple assignments

Irrelevant multiple assignments are circuitry-wise correct but do not obtain any additional information to the functional behavior as the functional blocks the transistor is in are of the same type. An example is a simple current mirror in a cascode current mirror (Fig. 3.10a). The simple current mirror does not obtain any additional information to the cascode current mirror. To avoid such irrelevant multiple assignments of transistors, the following rule is used:

$$\forall_{x_k \in \mathcal{X}_{min}} [\exists_{x_l \in \mathcal{X}} (x_k \subset x_l \wedge x_k.type = x_l.type)] \Leftrightarrow \ x_k \ \text{ is irrelevant} \tag{3.19}$$

$\mathcal{X}_{min}$ contains all functional blocks which are potentially irrelevant. These are all current mirrors, voltage and current biases.

False multiple assignments are circuitry-wise incorrect, e.g., two transistor stacks in a differential pair with current bias (Fig. 3.8). With suitable transistors connected at the drain of the differential pair, these false transistor stacks might form analog inverters. By suppressing the recognition of these transistor stacks, the false recognition of analog inverters is omitted.

3.1.3. Functional Blocks on Hierarchy Level 3

On HL 3 are the functional block types that form the amplification stages (HL 4) of an op-amp, i.e., transconductor, load and stage bias.

Transconductor

Function: A transconductor converts a voltage potential applied at its gates into an (amplified) current.

Structure: Fig. 3.11 shows the structural definition of a transconductor tc_k as well as examples. Two different types of transconductor exist, non-inverting and inverting.

A *non-inverting transconductor* tc_{ninv} consists of 1 or 2 differential pairs. We further define for tc_{ninv} three different types with three different structural descriptions:

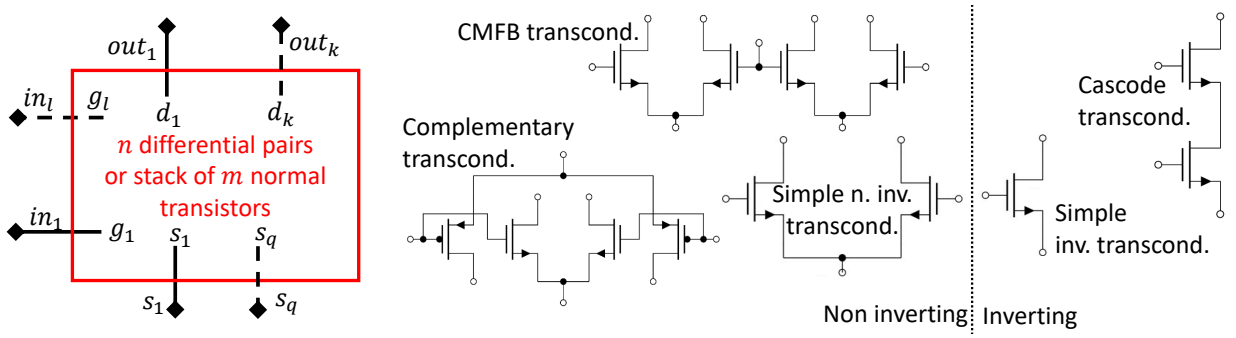


Figure 3.11.: Transconductor (transcond.) and examples (dashed lines: optional)

A *simple transconductor* tc_s is a transconductor consisting of one differential pair, having no gate connection to any other differential pair.

$$x_k = \{dp_k\} \wedge \nexists_{dp_l} \left(dp_k.g_y|_{y=1,2} \leftrightarrow dp_l.g_z|_{z=1,2}\right) \Leftrightarrow \quad x_k.type = tc_s \tag{3.20}$$

A *complementary transconductor* tc_c consists of two differential pairs with opposite doping and connected at both gates

$$x_k = \{dp_{k,1}, dp_{k,2}\} \wedge (dp_{k,1}.g_l \leftrightarrow dp_{k,2}.g_l)|_{l=1,2} \wedge dp_{k,1}.\Phi \neq dp_{k,2}.\Phi \Leftrightarrow x_k.type = tc_c \tag{3.21}$$

A *Common-Mode FeedBack (CMFB) transconductor* tc_{CMFB} consists of two differential pairs with equal doping connected at one of the two gates:

$$\begin{aligned} x_k = &\{dp_{k,1}, dp_{k,2}\} \wedge \exists!_{dp_{k,1}.g_m, dp_{k,2}.g_n} [dp_{k,1}.g_m \leftrightarrow dp_{k,2}.g_n] \wedge dp_{k,1}.\Phi = dp_{k,2}.\Phi \\ &\Leftrightarrow x_k.type = tc_{CMFB} \end{aligned} \tag{3.22}$$

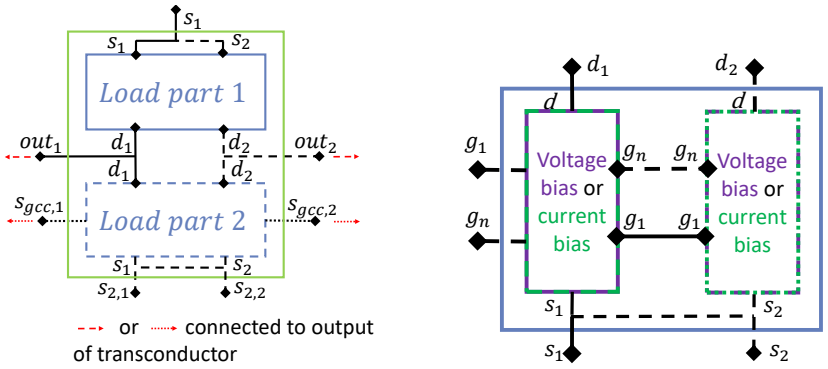
An *inverting transconductor* tc_{inv} consists of a transistor stack ts_k of m normal transistors, with the source $ts_k.s$ connected to a supply voltage rail. No gate-gate and gate-drain connection of the transistors in the stack is allowed. The gate of the bottom transistor in the stack $ts_k.g_1$ must be connected to the output of another transconductor $tc_v.out_p$ or to the output of a load $l_w.out_q$. The drain of the stack $ts_k.d$ must be connected to the output of a stage bias $b_{s,y}.out_z$.

$$\begin{aligned} x_k = &\{ts_k\} \wedge ts_k \subset \mathcal{X}_{nt} \wedge \text{net}(ts_k.s) \in \mathcal{N}_{supply} \wedge \forall_{nt_i, nt_j \in ts_k} (nt_i.g/d \nleftrightarrow nt_j.g/d) \\ &\wedge \left[\exists_{tc_v} [tc_v.out_p \leftrightarrow ts_k.g_1] \vee \exists_{l_w} [l_w.out_q \leftrightarrow ts_k.g_1]\right] \wedge \exists_{b_{s,y}} [b_{s,y}.out_z \leftrightarrow ts_k.d] \\ &\Leftrightarrow x_k.type = tc_{inv} \end{aligned} \tag{3.23}$$

The definition of the load and stage bias is given in the next two sections.

Load

Function: A load converts a current into a voltage. It influences the gain generated by the connected transconductor.



(a) A load functional block consists of one or two load parts

(b) Load part

Figure 3.12.: Load (dashed lines: optional)

Structure: Fig. 3.12 shows the structure of a load l_k. A load l_k consists of one or two load parts $l_{p,k,l}$ of type l_p. The parts are connected at their drains and have different substrate types. If a gate connected couple gcc_j is in one of the load parts, its sources $gcc_j.s_y$ are connected to the output of a transconductor tc_w of type tc_{ninv}. Otherwise, if no gate connected-couple is in one of the load parts, the outputs of the load $l_k.out_v$ are connected to $tc_w.out_v$.

$$\begin{aligned} x_k =& \{x_{k,1},..,x_{k,n} | n = |x_k| \wedge n \leq 2\} \wedge x_k \subseteq \mathcal{X}_{l_p} \wedge [n = 2 \rightarrow (x_{k,1}.\Phi \neq x_{k,2}.\Phi \\ & \wedge (x_{k,1}.d_l \leftrightarrow x_{k,2}.d_l)|_{l=1,2})] \wedge \left[\exists_{gcc_j \in x_k}[(gcc_j.s_y \leftrightarrow tc_z.out_y)|_{y=1,2,tc_z.type=tc_{ninv}}]\right. \\ & \left. \oplus (x_k.out_v \leftrightarrow tc_w.out_v)|_{v \leq |x_{k,1}|,\ tc_w.type=tc_{ninv}}\right] \Leftrightarrow x_k.type = l \end{aligned} \tag{3.24}$$

A single load part $l_{p,k,l}$ consists of one or two transistor stacks, which have gate-gate connections. A transistor stack in a load part is either of type voltage bias or of type current bias. If no gate connected-couple is in the load part, the sources must be connected. The doping of the transistor stacks is equal.

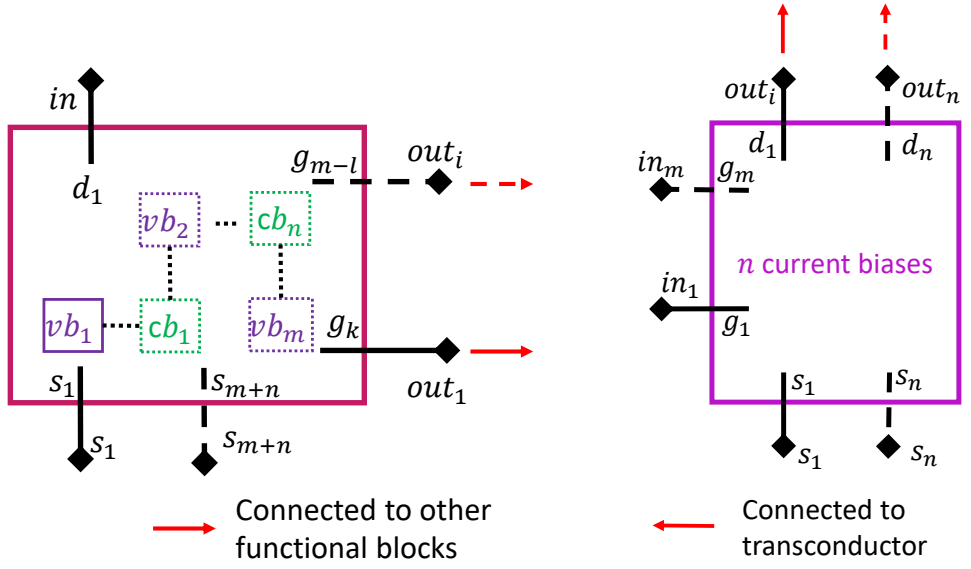
$$\begin{aligned} x_k =& \{x_{k,1},..,x_{k,n} | n = |x_k| \wedge n \leq 2\} \wedge x_k \subset (\mathcal{X}_{vb} \cup \mathcal{X}_{cb}) \wedge (x_{k,1}.g_l \leftrightarrow x_{k,n}.g_l)|_{l \leq |x_{k,1}|} \\ & \wedge \nexists_{gcc_l \in x_k}[x_{k,1}.s \leftrightarrow x_{k,n}.s] \wedge x_{k,1}.\Phi = x_{k,n}.\Phi \Leftrightarrow x_k.type = l_p \end{aligned} \tag{3.25}$$

Stage Bias

Function: A stage bias b_s supplies a transconductor with defined currents.

Structure: A stage bias b_s is a subtype of the type bias b (Fig. 3.13). Two different types exist.

A *bias with voltage output* b_v consists of m voltage and n current biases with $m > n$. Its input $b_{v,k}.in$ is the drain of a voltage bias. The outputs $b_{v,k}.out_1,..,b_{v,k}.out_i$ are gates of voltage biases, which are connected to gates of other functional blocks. All current biases in $b_{v,k}$ must have a drain-drain connection to a voltage bias with opposite doping and a



(a) A bias with voltage output is either a stage bias b_s or a circuit bias b_O

(b) A bias with current output is always a stage bias b_s

Figure 3.13.: Different types of biases (dashed lines: optional)

gate-gate connection to a voltage bias with equal doping in x_k.

$$\begin{aligned} x_k =& \{x_{k,1},..,x_{k,l} | l = |x_k|\} \wedge \{x_{k,1},..,x_{k,m}\} \subset \mathcal{X}_{vb} \wedge (l \neq m \rightarrow \{x_{k,l-m},..,x_{k,l}\} \subset \mathcal{X}_{cb}) \\ & \wedge \exists_{vb_{k,i} \in x_k} [vb_{k,i}.g_j \leftrightarrow x_z.g_y|_{x_z \in \mathcal{X} \setminus x_k}] \wedge \forall_{cb_{k,q} \in x_k} [cb_{k,q}.d \leftrightarrow vb_{k,v}.d \wedge vb_{k,v} \in x_k \\ & \wedge cb_{k,q}.\Phi \neq vb_{k,v}.\Phi \wedge (cb_{k,q}.g_s \leftrightarrow vb_{k,w}.g_s)|_{s \leq |cb_{k,q}|} \wedge vb_{k,w} \in x_k \wedge cb_{k,q}.\Phi = vb_{k,w}.\Phi] \\ & \Leftrightarrow x_k.type = b_v \end{aligned} \tag{3.26}$$

A bias with voltage output $b_{v,k}$ is a stage bias, iff it consists of exactly one voltage bias vb_k, which is connected with its drain $vb_k.d$ to the output of a transconductor of type inverting:

$$b_{v,k} = \{vb_k\} \wedge \exists_{tc_i} [tc_i.type = tc_{inv} \wedge vb_k.d \leftrightarrow tc_i.out] \Leftrightarrow b_{v,k}.type = b_s \tag{3.27}$$

In all other cases, if a bias with voltage output consists of more than one voltage bias or if it is not connected with its input pin to a transconductor, it is the circuit bias b_O (3.37).

A *bias with current output* b_c consists of n current biases. Its inputs $in_1, ..in_m$ are the gates of the current biases. The outputs $out_1, .., out_n$ are the drains of the current biases, which must be connected to the source or output of a transconductor.

$$\begin{aligned} x_k =& \{cb_{k,1},..,cb_{k,n} | n = |x_k|\} \wedge x_k \subset \mathcal{X}_{cb} \wedge \forall_{cb_{k,l} \in x_k} [cb_{k,l}.d \leftrightarrow (tc_z.s|_{tc_z.type = tc_{ninv}} \\ & \vee tc_z.out_y|_{tc_z.type = tc_{inv}})] \Leftrightarrow x_k.type = b_c \end{aligned} \tag{3.28}$$

A bias with current output is always a stage bias:

$$\forall_{b_k \in \mathcal{X}_b} [b_k.type = b_c \rightarrow b_k.type = b_s] \tag{3.29}$$

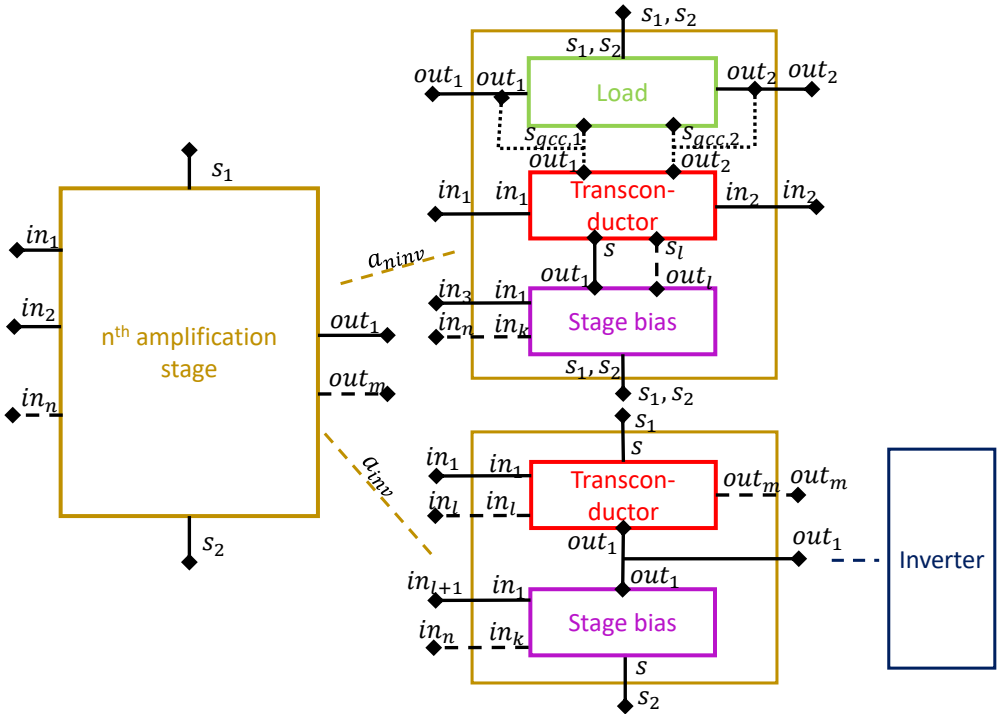


Figure 3.14.: Amplification stage (dashed lines: optional)

3.1.4. Functional Blocks on Hierarchy Level 4

The functional block types on HL 4 are the amplification stage, the circuit bias, the compensation and the load capacitor.

Amplification Stage

Function: Two functional types of amplification stage exist: the n^{th} amplification stage of an op-amp a_n and the Common-Mode FeedBack (CMFB) stage a_{CMFB}. The n^{th} amplification stage a_n amplifies the input signal for the n^{th} time. A common-mode feedback stage a_{CMFB} amplifies the output signals of a fully-differential op-amp while comparing them to a reference voltage and feeds them back to the amplifier.

Structure: Fig. 3.14 shows the general composition of the amplification stages of an op-amp. Every amplification stage has at least two inputs and one output. It has a source connection to both supply voltage rails. Two structural types of amplification stages exist, non-inverting and inverting amplification stages. Non-inverting amplification stages are the first stage of an op-amp and the CMFB stage. Inverting stages form the further stages of an op-amp ($n \geq 2$). In the following paragraphs, both types are described in detail.

A *non-inverting amplification stage* a_{ninv} consists of a transconductor tc_k, a load l_k and a stage bias $b_{s,k}$. The transconductor must be of type non-inverting and the stage bias must have a current output connected to the sources of the transconductor. The load outputs or the sources of a gate connected couple in the load must be connected to the outputs of the

transconductor.

$$x_k = \{tc_k, l_k, b_{s,k}\} \wedge tc_k.type = tc_{ninv} \wedge b_{s,k}.type = b_c \wedge \left[\exists_{gcc_i \in l_k}[(gcc_i.s_j \leftrightarrow tc_k.out_j)|_{j=1,2}] \oplus \forall_{l_k.out_v \in P_{l_k}}[l_k.out_v \leftrightarrow tc_k.out_w]\right] \wedge (tc_k.s_l \leftrightarrow b_{s,k}.out_l)|_{l \leq |tc_k|} \Leftrightarrow x_k.type = a_{ninv} \tag{3.30}$$

Non-inverting amplification stages are classified into three types, simple first stage a_s, complementary first stage a_c, and CMFB stage a_{CMFB}. The types differ in the transconductor types and the doping characteristics of their functional blocks.

A *simple first stage* a_s has a transconductor of type tc_s. The transconductor and the stage bias have the same substrate type, while the load has either the opposite doping or mixed doping.

$$a_{ninv} = \{tc_k, l_k, b_{s,k}\} \wedge tc_k.type = tc_s \wedge (tc_k.\Phi = b_{s,k}.\Phi) \in \{\Phi_n, \Phi_p\} \wedge tc_k.\Phi \neq l_k.\Phi \Leftrightarrow a_{ninv}.type = a_s \tag{3.31}$$

A *complementary first stage* a_c has a transconductor of type tc_c. Transconductor, load and stage bias have all mixed doping.

$$a_{ninv} = \{tc_k, l_k, b_{s,k}\} \wedge tc_k.type = tc_c \wedge tc_k.\Phi = b_{s,k}.\Phi = l_k.\Phi = \Phi_u \Leftrightarrow a_{ninv}.type = a_c \tag{3.32}$$

A *common-mode feedback stage* a_{CMFB} has a transconductor of type tc_{CMFB}. Transconductor and stage bias have the same substrate type, while the load has opposite doping.

$$a_{ninv} = \{tc_k, l_k, b_{s,k}\} \wedge tc_k.type = tc_{CMFB} \wedge (tc_k.\Phi = b_{s,k}.\Phi) \in \{\Phi_n, \Phi_p\} \wedge tc_k.\Phi \neq l_k.\Phi|_{l_k.\Phi \in \{\Phi_n, \Phi_p\}} \Leftrightarrow a_{ninv}.type = a_{CMFB} \tag{3.33}$$

An *inverting amplification stage* a_{inv} consists of a transconductor tc_k and a stage bias $b_{s,k}$ with opposite doping connected at their outputs. The transconductor must be of type inverting. The stage bias consists either of one voltage bias or one current bias.

$$a_k = \{tc_k, b_{s,k}\} \wedge tc_k.type = tc_{inv} \wedge b_{s,k}.type \in \{b_c, b_v\} \wedge |b_{s,k}| = 1 \wedge tc_k.out_1 \leftrightarrow b_{s,k}.out_1 \wedge tc_k.\Phi \neq b_{s,k}.\Phi \Leftrightarrow a_k.type = a_{inv} \tag{3.34}$$

Note that all inverting stages can occur twice in an op-amp, e.g., two second stages. It appears when the op-amp is fully-differential or a symmetrical op-amp (Fig. 3.2, with $a_{2,1}, a_{2,2}$).

Differentiating between an inverting stage with a stage bias with current output and an inverting stage with a stage bias with voltage output allows to give a more precise definition of inverting amplification stages:

An *inverting stage with stage bias with current output* a_{inv_c} must also fulfill the type definition of an analog inverter to be a valid inverting amplification stage:

$$a_{inv,k} = \{tc_{inv,k}, b_{s,k}\} \wedge b_{s,k}.type = b_c \wedge a_{inv,k}.type = inv \Leftrightarrow a_{inv,k}.type = a_{inv_c} \tag{3.35}$$

Note that not all analog inverters are inverting stages, as the specifications for the transconductor (3.23) and stage bias (3.28) must be fulfilled.

An *inverting stage with stage bias with voltage output* a_{inv_v} only occurs in symmetrical OTAs, which are op-amps with a characteristic first stage and two second stages (Fig. 3.2). The first stage a_1 must be a simple first stage a_s having a load consisting of two voltage biases $vb_{l,1}, vb_{l,2}$ with the same doping. Both voltage biases must have a gate-gate connection to a transconductor in a second stage of type a_{inv}. One second stage a_z must be of type a_{inv_c}. The other second stage a_y must have a voltage bias as stage bias, which has a gate-gate connection to the stage bias of a_z:

$$\begin{aligned} a_{inv,k} = \{tc_{inv,k}, b_{s,k}\} \wedge b_{s,k}.type = b_v \wedge \exists_{a_1} \Big[& a_1.type = a_s \wedge \exists_{l_1 \in a_1} \big[|l_1| = 1 \\ & \wedge \exists_{l_{1,1} \in l_1} [\{vb_{l,1}, vb_{l,2}\} \subseteq l_{1,1} \wedge (vb_{l,m}.g_1 \leftrightarrow tc_{inv,k}.in_1 \wedge vb_{l,n}.g_1 \leftrightarrow a_i.in_1)|_{(m=1 \wedge n=2) \vee (m=2 \wedge n=1)} \\ & \wedge a_i.type = a_{inv_c} \wedge (b_{s,k}.out_v \leftrightarrow b_i.in_v|_{b_i \in a_i})] \big] \Big] \Leftrightarrow a_{inv,k}.type = a_{inv_v} \end{aligned} \tag{3.36}$$

Circuit Bias

Function: The circuit bias b_O supplies all functional blocks of the type amplification stage a with voltages specified by b_O.

Structure: The circuit bias b_O has the structure of a bias with voltage output (3.26) (Fig. 3.13a). It contains all current biases and voltage biases that are not part of an amplification stage.

$$x_k = \{x_{k,1}, .., x_{k,n} | n = |x_k|\} \wedge x_k = [(\mathcal{X}_{vb} \cup \mathcal{X}_{cb}) \setminus \mathcal{X}_a] \wedge x_k.type = b_v \Leftrightarrow x_k.type = b_O \tag{3.37}$$

Compensation Capacitor

Function: A compensation capacitor $c_{C,k}$ increases the stability of an op-amp.

Structure: A compensation capacitor $c_{C,k}$ is connected between the outputs of two different amplification stages a_j, a_v:

$$x_k = \{cap_k\} \wedge [cap_k.p_1 \leftrightarrow a_j.out_i \wedge cap_k.p_2 \leftrightarrow a_v.out_w]|_{a_j \neq a_v} \Leftrightarrow \; x_k.type = c_C \tag{3.38}$$

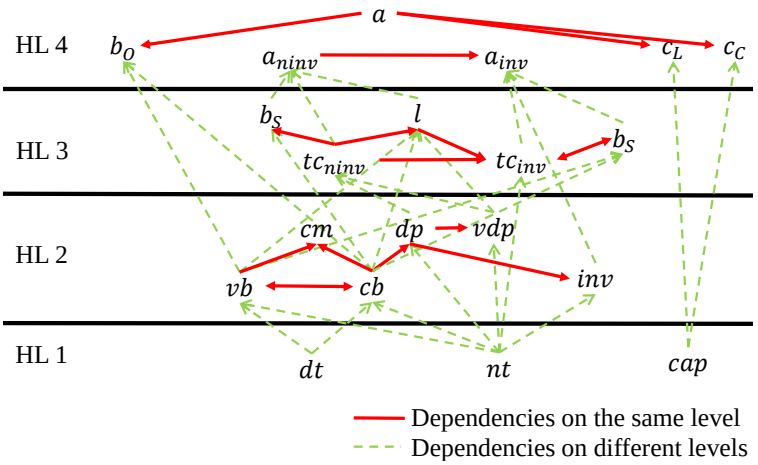


Figure 3.15.: Dependency graph of the functional blocks of an op-amp (see Fig. 3.1)

Load Capacitor

Function: The load capacitor $c_{L,k}$ represents the capacitive load the op-amp is able to drive in its application.

Structure: A load capacitor $c_{L,k}$ is connected between an output of the highest (n-th) amplification stage a_n and ground:

$$x_k = \{cap_k\} \wedge cap_k.p_1 \leftrightarrow a_n.out_j \wedge \text{net}(cap_k.p_2) = GND \Leftrightarrow \; x_k.type = c_L \tag{3.39}$$

3.2. Functional Block Analysis

The functional block analysis recognizes all functional blocks in an op-amp netlist based on the definitions given in Sec. 3.1.1 - 3.1.4. Fig. 3.15 shows the dependency graph that arises from these definitions. Note that bidirectional dependencies exist. Therefore, complex algorithms are required for the functional block analysis.

3.2.1. Hierarchy Level 1

On hierarchy level 1, the definitions of the functional block types are independent of each other. Therefore, the definitions given in Sec. 3.1.1 can be used for recognition without any restriction to their order.

3.2.2. Hierarchy Level 2

On HL 2, every recognition of a functional block type depends on another functional block on the same level (3.10) - (3.18). Voltage bias and current bias are at the starting point of the dependency graph being bidirectionally dependent.

Alg. 1 presents the recognition algorithm. To resolve all dependencies, the algorithm starts with the recognition of transistor stacks (3.9), which are auxiliary blocks independent of all

Algorithm 1 Recognition of functional block types on hierarchy level 2

Require: $\mathcal{X} = \mathcal{X}_{nt} \cup \mathcal{X}_{dt}$
1: $\mathcal{X}_{vb} := \{\}$ //At the beginning the set of voltage biases is empty
2: $\mathcal{X}_{cb} := \{\}$ //At the beginning the set of current biases is empty
3: $\mathcal{X}_{ts_n}$:=findAllTransistorStacksNMOS($\mathcal{X}$) //Definition (3.9)
4: $\mathcal{X}_{ts_p}$:=findAllTransistorStacksPMOS($\mathcal{X}$) //Definition (3.9)
5: **repeat**
6: **for all** $ts_k \in [\mathcal{X}_{ts_j}|_{j\in\{n,p\}} \setminus (\mathcal{X}_{vb} \cup \mathcal{X}_{cb})]$ **do**
7: **if** $\exists_{ts_l \in \mathcal{X}_{ts_j}} \left[ts_k.d \leftrightarrow ts_l.g_m \wedge \forall_{ts_k.g_v}[ts_k.g_v \leftrightarrow ts_l.g_w]\right]$ **then**
8: **if** $\forall_{ts_l.g_m}[ts_l.g_m \leftrightarrow (ts_k.g_m \vee ts_k.d)] \wedge$ $\nexists_{ts_i \in \mathcal{X}_{ts_j}}[ts_i.g_n \leftrightarrow ts_l.d]$ **then**
9: $\mathcal{X}_{vb} := \mathcal{X}_{vb} \cup \{ts_k\}$
10: $\mathcal{X}_{cb} := \mathcal{X}_{cb} \cup \{ts_l\}$
11: **else if** $ts_l \in \mathcal{X}_{cb}$ **then**
12: $\mathcal{X}_{vb} := \mathcal{X}_{vb} \cup \{ts_k\}$
13: **end if**
14: **end if**
15: **end for**
16: **until** no new voltage or current bias was found
17: $\mathcal{X}_{cm}$:= findAllCurrentMirrors($\mathcal{X}_{vb}, \mathcal{X}_{cb}$) //Definition (3.12)
18: $\mathcal{X}_{dp}, \mathcal{X}_{vdp}$:= findAllDifferentialPairs($\mathcal{X}_{nt}, \mathcal{X}_{cb}$) //Definitions (3.13), (3.15)
19: $\mathcal{X}_{inv}$:= findAllAnalogInverters($\mathcal{X}_{ts_n}, \mathcal{X}_{ts_p}$) //Definition (3.18)
20: $\mathcal{X} := \mathcal{X} \cup \mathcal{X}_{vb} \cup \mathcal{X}_{cb} \cup \mathcal{X}_{cm} \cup \mathcal{X}_{dp} \cup \mathcal{X}_{vdp} \cup \mathcal{X}_{inv}$
21: $\mathcal{X}$:= deleteIrrelevantStructures($\mathcal{X}$) //Definition (3.19)
22: **return** $\mathcal{X}$

functional block types of level 2. On basis of the recognized transistor stacks, the current and voltage biases are recognized. For every substrate type, a set of transistor stacks $\mathcal{X}_{ts_j}|_{j\in\{n,p\}}$ is created (Line 3, 4). The algorithm iterates over both sets. It searches for two transistor stacks ts_k, ts_l, which are connected at their gates and also have a drain-gate connection $ts_k.d \leftrightarrow ts_l.g_m$ (Line 7). This corresponds to the definition of a voltage bias (3.10), taking ts_k as voltage bias connected to a current bias ts_l .

It is checked if ts_l has all the gate-drain and gate-gate connections to ts_k needed for a current bias (Line 8), and if it is not connected with its drain to any gate of a transistor stack in $\mathcal{X}_{ts_j}$. This corresponds to the definition of a current bias (3.11). If this is the case, the ts_k and ts_l are *primary voltage and current biases*, i.e., they have all gate-gate and gate-drain connection to each other. Another case is that ts_l is an already identified current bias (Line 11). In this case, the primary voltage and current bias were already recognized and a *secondary voltage bias* ts_k is recognized, e.g., a voltage bias which biases the uppermost gate of a wide-swing cascode current mirror.

Note that the last part of the voltage bias definition (3.10) is not checked during recognition as this is always ensured by using a valid op-amp topology.

The repetitive iteration over all recognized transistor stacks is needed to ensure that all secondary voltage biases are found. After finding all voltage and current biases, the current mirrors and differential pairs are found using their definitions. The analog inverters are the last functional blocks of HL 2 that are recognized to omit their false recognition in differential

pairs. With (3.19), all irrelevant functional blocks are deleted from the set of all functional blocks.

3.2.3. Hierarchy Level 3 - 4

We combine the recognition of the functional block types of HL 3 and HL 4 as this simplifies the recognition process and resolves the bidirectional dependency of the inverting transconductor tc_{inv} and its stage bias b_s.

The algorithm (Alg. 2) starts by recognizing all non-inverting transconductors. They are the only functional blocks on HL 3 that are independent of any other functional block (3.30).

In the next step, the loads are recognized by Alg. 3. The algorithm does not use the exact definition of the load (3.24). The advantage is that the recognition does not depend on recognized voltage or current biases as the definition of the load (3.25) does. Often external voltage biases are used to bias the load. In this case, current biases as load are not recognized. Alg. 3 is more general and uses transistor stacks for recognition. It first searches for the nets to which the load parts are connected (Line 4 - 9). This is either the drain net of a gate connected couple $gcc_l.d$ or if no gcc_l exists, the drain nets of differential pairs in $tc_{ninv,k}$. At these nets, the algorithm searches for the transistor stacks forming the load (Line 12 - 19). Note that a gate connected couple if recognized, is part of a load while a differential pair is not.

After the recognition of the loads, the stage biases of non-inverting stages are recognized by their definition (3.29). With the non-inverting transconductor, loads and stage biases, the non-inverting stages are recognized.

In the next step, inverting stages are recognized. For recognition, we use that an inverting stage is also a functional block of type analog inverter, iff its stage bias is of type current bias (3.35). This resolves the bidirectional dependency of an inverting transconductor $tc_{inv,k}$ and its bias $b_{s,inv,k}$.

The algorithm iterates over all recognized analog inverters. For every analog inverter, it is checked if one of its stacks $ts_{k,1}$ is connected with its first gate $ts_{k,1}.g_1$ to the output of an already recognized stage a_i. This corresponds to the definition of an inverting transconductor $tc_{inv,k}$ (3.23). The output of a stage is either the output of its load or the output of its transconductor (Fig. 3.11). The other transistor stack $ts_{k,2}$ in the analog inverter must be of type current bias and thus is the stage bias (3.29) of the inverting stage.

After finding the first inverting stage with current bias as stage bias, it is searched for an inverting stage with voltage bias as stage bias (3.36). This type of inverting stage is only part of an op-amp if the first stage fulfills the criteria of a symmetrical op-amp (Line 17). It is sufficient to search for a transistor stack ts_i connected with its gate $ts_i.g_1$ to one of the voltage biases in the first stage load. ts_i must be connected with its drain $ts_i.d$ to a voltage bias drain $vb_j.d$. The voltage bias vb_j must be connected with its gates to the current bias of the already recognized inverting stage $a_{inv,1}$.

As inverting stages might be connected to other inverting stages, e.g, a third stage connected to a second stage, it is repeatedly iterated over the set of analog inverters until no new stage is found. Thus, also multi-stage op-amps are supported by this method. Multi-stage op-amps usually comprise multiple inverting stages. Some of them may be connected as frequency

Algorithm 2 Recognition of functional block types on the hierarchy levels 3-4

Require: $\mathcal{X}$

1: //Searching for non-inverting stages
2: $\mathcal{X}_{tc_{ninv}}$:=findAllNonInvertingTranscond($\mathcal{X}_{dp}, \mathcal{X}_{vdp}$)//Definitions (3.20),(3.21),(3.22)
3: $\mathcal{X}_{l}$:=findAllLoads($\mathcal{X}_{tc_{ninv}}, \mathcal{X}$) //Algorithm 3,
4: $\mathcal{X}_{b_{s,ninv}}$:=findStageBiases($\mathcal{X}_{tc_{ninv}}, \mathcal{X}_{cb}$) //Definition (3.29)
5: $\mathcal{X}_{a_{ninv}}$:=findAllNonInvertingStages($\mathcal{X}_{tc_{ninv}}, \mathcal{X}_{l}, \mathcal{X}_{b_{s,ninv}}$)//Definitions (3.31),(3.32),(3.33)
6: $\mathcal{X} := \mathcal{X} \cup \mathcal{X}_{tc_{ninv}} \cup \mathcal{X}_{l} \cup \mathcal{X}_{b_{s,ninv}} \cup \mathcal{X}_{a_{ninv}}$
7: //Searching for inverting stages
8: $\mathcal{X}_{a_{inv}} = \{\}$
9: **repeat**
10: **for all** $inv_k \in \mathcal{X}_{inv}$ **do**
11: **if** $\exists_{ts_{k,1} \in inv_k}[\exists_{a_i \in \mathcal{X}_a} a_i.out_j \leftrightarrow ts_{k,1}.g_1] \wedge \exists_{ts_{k,2} \in inv_k}(ts_{k,2}.type = cb)$ **then**
12: $tc_{inv,k} = ts_{k,1}$
13: $b_{s,inv,k} = ts_{k,2}$
14: $a_{inv,k} = \{tc_{inv,k}, b_{s,inv,k}\}$
15: $\mathcal{X} := \mathcal{X} \cup \{tc_{inv,k}, b_{s,inv,k}, a_{inv,k}\}$
16: **end if**
17: **if** $|\mathcal{X}_{a_{inv}}| = 1 \wedge \exists_{a_1}\left[a_1.type = a_s \wedge \exists_{l_1 \in a_1}[|l_1| = 1 \wedge \exists_{l_{1,1} \in l_1}(\{vb_{l,1}, vbl_2 \subseteq l_{1,1})]]\right]$ **then**
18: **if** $\exists_{ts_i}[ts_i.g_1 \leftrightarrow (vb_{l,1}.g_1 \vee vb_{l,2}.g_1) \wedge ts_i.d \leftrightarrow vb_j.d \wedge (vb_j.g_q \leftrightarrow a_{inv,1}.g_q|_{a_{inv,1} \in \mathcal{X}_{a_{inv}}})|_{q \leq |vb_j|}]$ **then**
19: $tc_{inv,i} = ts_i$
20: $b_{s,inv,v} = vb_j$
21: $a_{inv,v} = \{tc_{inv,i}, b_{s,inv,v}\}$
22: $\mathcal{X} := \mathcal{X} \cup \{tc_{inv,i}, b_{s,inv,v}, a_{inv,v}\}$
23: **end if**
24: **end if**
25: **end for**
26: **until** no new stage is found
27: $\mathcal{X}_{b_O}$:= findCircuitBias($\mathcal{X}_{cb} \cup \mathcal{X}_{vb}, \mathcal{X}_a$) //Definition (3.37)
28: $\mathcal{X}_{c_C}$:= findCompensationCapacitors($\mathcal{X}_{cap}$,$\mathcal{X}_a$) //Definition (3.38)
29: $\mathcal{X}_{c_L}$:= findLoadCapacitors($\mathcal{X}_{cap}$,$\mathcal{X}_a$) //Definition (3.39)
30: $\mathcal{X} := \mathcal{X} \cup \mathcal{X}_{b_O} \cup \mathcal{X}_{c_C} \cup \mathcal{X}_{c_L}$
31: **return** $\mathcal{X}$

Algorithm 3 findAllLoads($\mathcal{X}_{tc_{ninv}}, \mathcal{X}$)

Require: $\mathcal{X}_{tc_{ninv}}$; $\mathcal{X}$
1: $\mathcal{X}_l := \{\}$
2: **for all** $tc_{ninv,k} \in \mathcal{X}_{tc_{ninv}}$ **do**
3: $\mathcal{N} := \{\}$ // Set of nets the load parts are connected to is empty
4: **if** $\exists_{dp_l \in tc_{ninv,k}} \{dp_l\} \subset vdp_l$ **then**
5: $gcc_l = vdp_l \setminus \{dp_l\}$
6: $\mathcal{N} := \mathcal{N} \cup \{\text{net}(gcc_l.d_1), \text{net}(gcc_l.d_2)\}$
7: **else**
8: $\mathcal{N} := \mathcal{N} \cup \{\text{net}(tc_{inv,k}.out_1), \text{net}(tc_{inv,k}.out_2), ..\}$
9: **end if**
10: $l_{p,n} := \{\}$//The load part containing NMOS transistors is empty
11: $l_{p,P} := \{\}$//The load part containing PMOS transistors is empty
12: **for all** $n_j \in \mathcal{N}$ **do**
13: **if** $\exists_{ts_m} \left[ts_m.\Phi = \Phi_n \wedge \text{net}(ts_m.d) = n_j \wedge [ts_m.s \leftrightarrow (GND \vee tc_{ninv,k}.out_z)]\right]$ **then**
14: $l_{p,n} := l_{p,n} \cup \{ts_m\}$
15: **end if**
16: **if** $\exists_{ts_n} \left[ts_n.\Phi = \Phi_p \wedge \text{net}(ts_n.d) = n_j \wedge [ts_n.s \leftrightarrow (VDD \vee tc_{ninv,k}.out_z)]\right]$ **then**
17: $l_{p,p} := l_{p,p} \cup \{ts_n\}$
18: **end if**
19: **end for**
20: $\mathcal{X}_l := \mathcal{X}_l \cup \{l_{p,p}, l_{p,n}\}$
21: **end for**
22: **return** $\mathcal{X}_l$

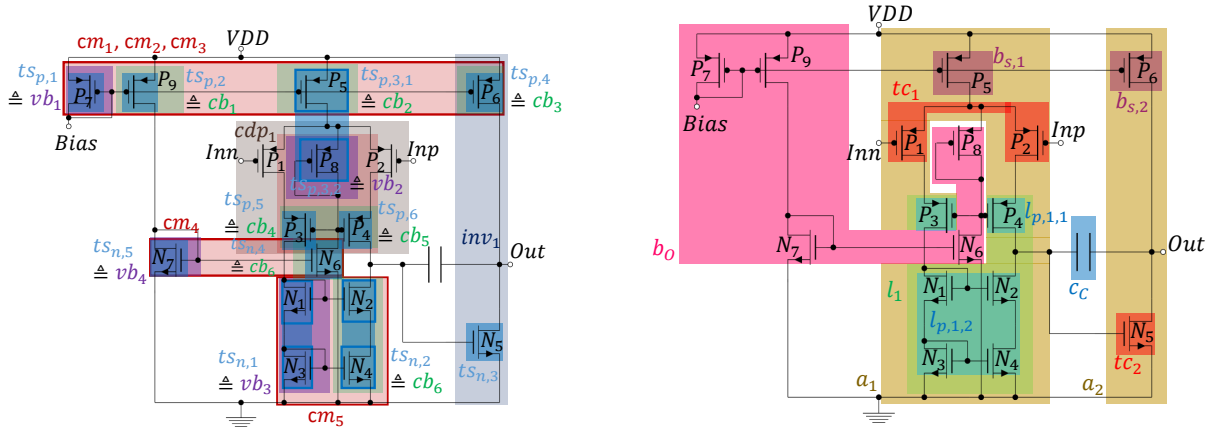
compensation in feedback loops [76, 77]. However, as they have the characteristic structure of an inverter and as the gate of one of the transistors in the transconductor is connected to the output of the previous stage, they are unambiguously identifiable.

After finding all amplification stages of an op-amp, the circuit bias, the compensation capacitors and the load capacitors are recognized using their definitions.

3.3. Experimental Results

In this section, experimental results of the functional block analysis are presented. The algorithms presented in Sec. 3.2 are illustrated on the example of a telescopic two-stage op-amp (Fig. 3.16). Furthermore, the results of the functional block analysis on four different circuits are discussed: a symmetrical op-amp (Fig. 3.2), a folded-cascode op-amp with Common-Mode FeedBack (CMFB) (Fig. 3.17), a complementary op-amp (Fig. 3.18) and a three-stage op-amp (Fig. 3.19). Overall more than 4000 different op-amp topologies have been successfully analyzed.

The computational cost of the algorithm is very low. The runtime for every topology is in the area of milliseconds. The time needed to add a new functional block to the algorithm depends on the uniqueness of its structures. It is very low and in the area of a few hours, if many of the already implemented functional blocks are reused. More advanced functional



(a) Level 2 (Alg. 1)

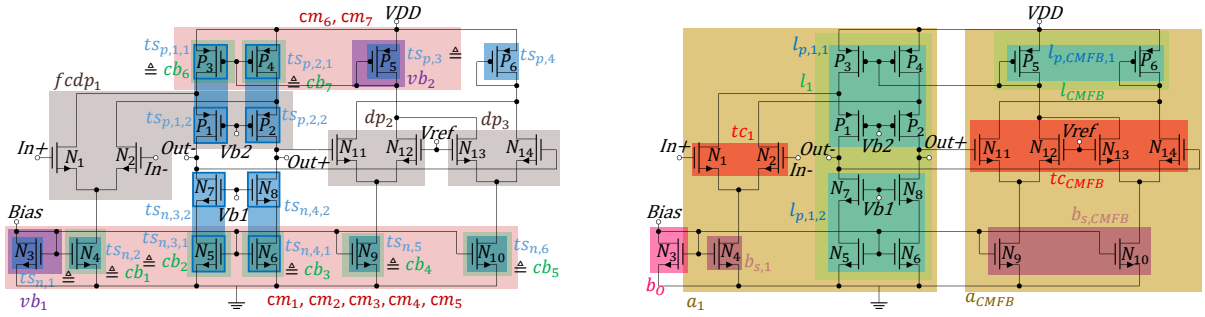
(b) Level 3 - 4 (Alg. 2)

Figure 3.16.: Blocks recognized during functional block analysis in a telescopic two-stage op-amp

blocks, e.g., frequency compensation methods as described in [78,79], need a day integration time.

Telescopic op-amp (HL 2): Fig. 3.16a shows all functional blocks recognized on level 2 with Alg. 1. Alg. 1 first searches for all transistor stacks in the circuit sorted according to their substrate type. All transistor stacks are marked in dark blue. Transistor stacks including transistors of the differential pair (P_1, P_2) are not shown as they are not valid (Sec. 3.1.2). Every transistor in the circuit is a transistor stack by itself. That means for the three transistor stacks consisting of two transistors, every transistor in these stacks is also a transistor stack by itself. The algorithm checks which of the transistor stacks are voltage and current biases. Note that, for P_5, P_8, the transistor stacks of the individual transistors are classified as a current bias and voltage bias, respectively. P_5 has a gate connection to the voltage bias $vb_1(P_7)$ and therefore fulfills the definition of a current bias. P_8 has a gate-drain connection to itself and therefore is a voltage bias connected with its gate to the gates of the current biases $cb_4(P_3)$ and $cb_5(P_4)$. The other two-transistor stacks, N_1, N_3 and N_2, N_4, form a current mirror. The transistor stacks of the individual transistors therefore are irrelevant. Note that vb_2, cb_4 and vb_2, cb_5 do not form current mirrors as their sources are not connected to the same net. All voltage and current biases in this circuit are primary. All voltage biases have their needed gate-drain connection (3.10) by themselves, such that no secondary voltage biases are needed to establish a drain-gate connection of a current bias gate. Alg. 1 ends with the recognition of the cascode differential pair cdp_1 (3.17) and the analog inverter inv_1 (3.18).

Telescopic op-amp (HL 3 - HL 4): Fig. 3.16b shows all functional blocks recognized on levels 3 and 4 with Alg. 2. The algorithm first recognizes the differential pair as simple non inverting first stage transconductor tc_1. For the load recognition, the drain nets of P_3, P_4 are used. P_3, P_4 form the gate connected couple part of the cascode differential pair cdp_1. Two load parts are found: The gate connected couple forms the load part with p-doping. The load part with n-doping consists of the current mirror $cm_5 = \{ts_{n,1}, ts_{n,2}\}$. As stage bias, the current bias cb_2 is found as it is connected with its drain to the source of tc_1. With it, all parts of the non-inverting first stage a_1 are recognized. In the next step, Alg. 2 checks if the inverter recognized on level 2 is a second stage. This is true, as P_6 was recognized as current



(a) Functional blocks on level 2

(b) Functional blocks on level 3 - 4

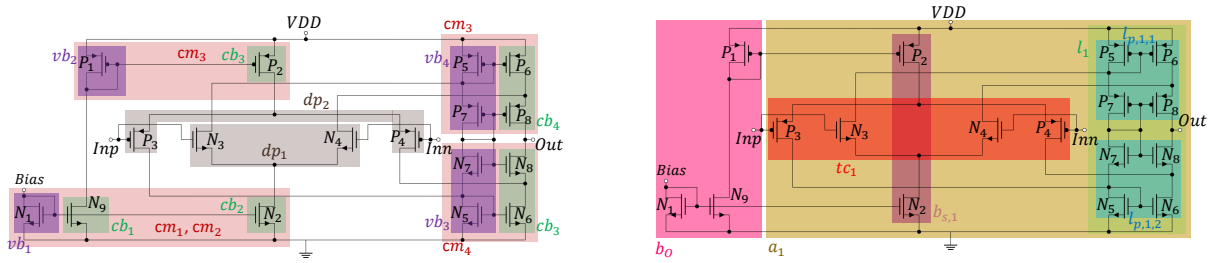
Figure 3.17.: Folded-cascode op-amp with CMFB

bias cb_3 and $ts_{n,3}(N_5)$ is connected with its gate to one output of the first stage. Because the first stage does not fulfill the criteria of a symmetrical op-amp, it is not searched for an inverting stage with voltage bias as stage bias. The voltage and current biases which are not part of the two amplification stages are recognized as circuit bias b_O. The recognition ends by identifying the capacitor in the circuit as compensation capacitor.

Symmetrical op-amp with an additional inverting stage: Fig. 3.2 shows the results of the functional block analysis for a symmetrical op-amp with an additional inverting stage a_3. During the recognition of the inverting stages, first inv_2 is recognized as inverting second stage $a_{2,1}$. Afterwards, $a_{2,2}$ is recognized, a_1 fulfills the condition of a first stage of a symmetrical op-amp and with $a_{2,1}$, the needed second stage with current bias as stage bias is given (3.36). The transistor stack $\{nt_4(P_3), nt_6(P_5)\}$ fulfills the definition of a non-inverting transconductor $tc_{2,2}$ (3.23). vb_2 is the stage bias with voltage output. The gate-gate connection to the stage bias of $a_{2,1}$ is given. Note that for a correct identification of a_3 the exact definition of an inverting stage must be used (3.34). As P_7 has a gate connection to one output of the first stage, a drain of the differential pair, it is considered as possible inverting transconductor of another second stage. As for this transconductor, no stage bias exists because N_6 is not of type bias, it is not recognized as transconductor. Instead, after $a_{2,1}$ is recognized, N_6 is identified as inverting transconductor, because it has a connection to an output of $tc_{2,2}$. P_7 is its stage bias.

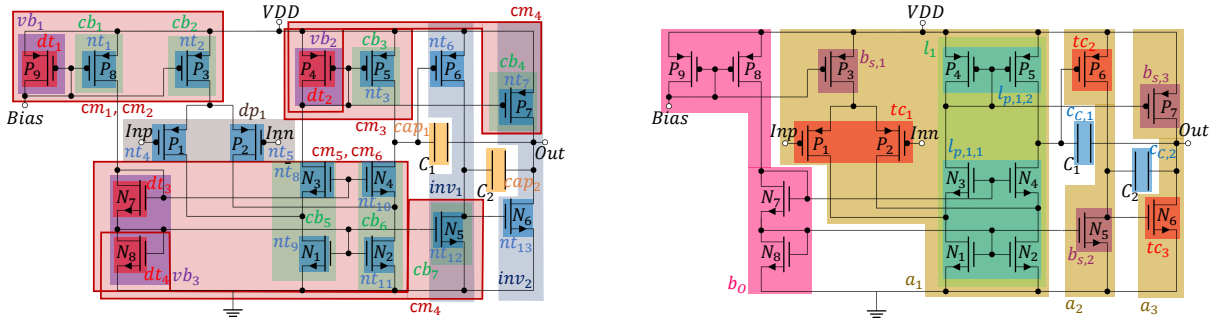
Folded-cascode op-amp with Common-Mode FeedBack (CMFB): Fig. 3.17 shows the results of the functional block analysis for a folded-cascode op-amp with CMFB. The four load transistors P_1, P_2, N_7, N_8 are externally biased by the pins $Vb1$ and $Vb2$ respectively. Hence, these four transistors are not recognized as current biases (Fig. 3.17a) as they are not connected with their gates to a voltage bias. However with Alg. 3, they are still recognized as part of the load (Fig. 3.17b). P_6 is also neither identified as current bias nor as voltage bias as it does not have any gate-gate connections to a another transistor with the same doping. However, it fulfills the functional definition of a load as it has a drain-drain connection to tc_{CMFB}. Therefore, it is recognized with Alg. 3 correctly even if it does not fulfill the definition of a load entirely (3.24).

Complementary op-amp: The complementary op-amp in Fig. 3.18 has a complementary first stage a_1 with a transconductor tc_1 of type complementary. Its stage bias consists of one current bias for each doping connected to the source of the differential pair in tc_1 with the same doping. For the recognition of the load, P_7, P_8 and N_7, N_8 are treated like two



(a) Functional blocks on HL 2 (b) Functional blocks on HL 3 - HL 4

Figure 3.18.: Complementary op-amp



(a) Functional blocks on HL 1 - HL 2 (b) Functional blocks on HL 3 - HL 4

Figure 3.19.: Three-stage op-amp

gate connected couples as they have the same functional definition and only differ from the structural definition by an additional gate-drain connection (3.14).

Three-stage op-amp: Fig. 3.19 shows the results of the functional block analysis for a three-stage op-amp. Two analog inverters inv_1, inv_2 are recognized on HL 2 (Fig. 3.19a). One of the transistors N_5 (inv_1), P_7 (inv_2) is identified to be part of a current mirror. The other transistor P_6 (inv_1), N_6 (inv_2) is not part of any additional functional block on HL 2. After recognizing the first and second stage with Alg. 2, the third stage is recognized (Fig. 3.19b). With the gate of N_6 connected to the drain of N_5, the input of third stage is connected to the output of the second stage (3.34). P_7 is an identified current bias. Hence, inv_2 fulfills all criteria of a third stage. Alg. 2 ends by identifying the compensation capacitors $c_{c,1}, c_{c,2}$. Both capacitors are connected between the output of two stages, $c_{c,1}$ between the output of first and third stage, $c_{c,2}$ between second and third stage. Hence, both capacitors fulfill (3.38).

4. Sizing via Functional Block Modeling

This chapter presents a new the equation-based sizing method. Different to the state of the art, the sizing method is topology independent. For every topology, the analytical equations are automatically set up and solved. Every inserted topology is analyzed by the functional block decomposition method described in Chapter 3. The behavior models belonging to the recognized functional blocks are automatically set up. This is achieved by a hierarchical performance equation library (HPEL) storing the behavior model of every functional block in a topology independent formalized way. The equation-based model is solved using Constraint Programming. A problem specific branching heuristic is used to make Constraint Programming a reliable solver for the equation-based initial sizing problem of analog circuit.

In the following section, an overview of the automatic equation-based initial sizing method is given (Sec. 4.1). Sec. 4.2 presents the hierarchical performance equation library. Algorithms to set up the equation model of the circuit automatically are presented in Sec. 4.3. Sec. 4.4 gives an overview of Constraint Programming. The adoptions we made to Constraint Programming to make it a reliable solver for the sizing problem of analog circuits are described in Sec. 4.5. Experimental results obtained with the sizing method are given in Sec. 4.6.

4.1. Overview of the Automatic Equation-Based Initial Sizing Method

Fig. 4.1 gives an overview of the automatic equation-based initial sizing method. It has as input the circuit netlist, the process parameters, the performance requirements and the external supplies. It outputs the devices sizes and the expected performance behavior.

Three steps are performed to obtain the devices sizes:

1. **Functional block analysis**: The functional blocks of the op-amp are derived from the circuit netlist. This step is automated by the functional block analysis in Chapter 3.
2. **Set-up of the equation-based circuit performance model**: With the information gained by the functional block analysis, the equations describing the op-amp behavior are automatically set up (Sec. 4.3). A library (HPEL) was developed (Sec. 4.2) storing the equations describing the behavior of the different functional blocks in op-amp hierarchically. For every performance equation, constraints are derived, restricting the performance variables to fulfill their required values.
3. **Solving**: The equation-based initial sizing problem generated in Step 2 combines sizing with optimization in a single problem. We use Constraint Programming (Sec. 4.4) to solve it. The solver has, additionally to the constraints and equations of Step 2, the process parameters, external supplies and performance requirements as input. A new problem-specific dynamical branching heuristic was developed to make Constraint Programming an efficient and reliable solver (Sec. 4.5).

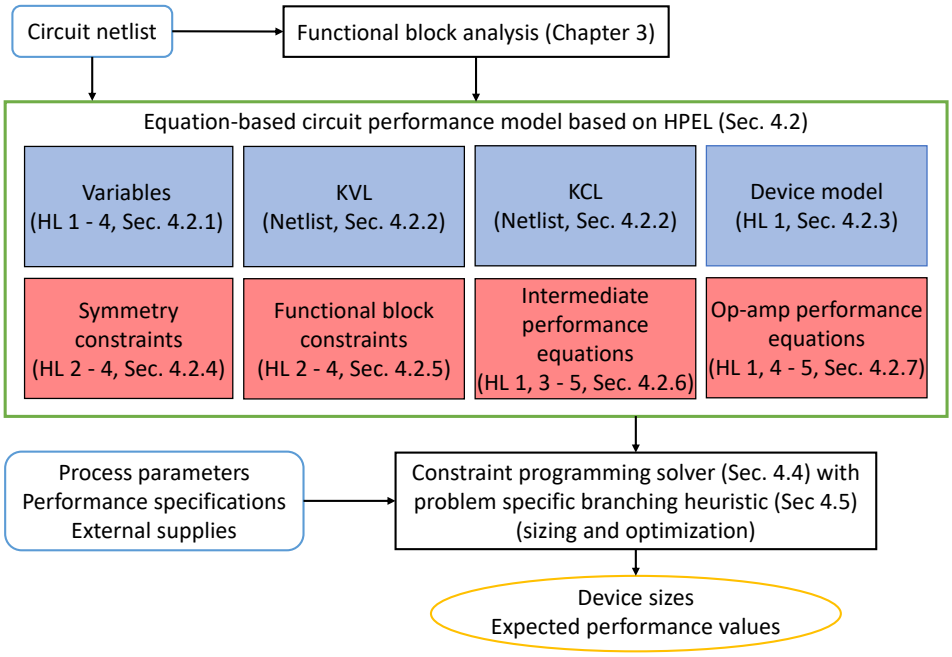


Figure 4.1.: Overview of the automatic equation-based initial sizing method

Through the topology independence gained by the functional block analyses, by the Hierarchical Performance Equation Library (HPEL), and by the dynamical branching heuristic of the constraint programming solver, the sizing method supports many different op-amp topologies including hundreds of variations, e.g., in load and bias. In the sizing process, the method abolishes the usual distinction between simulation and optimization ("simulation in a loop") and combines both parts to one process, similar to [16]. This presents additional challenges to the constraint optimization algorithm. In the following sections, the hierarchical performance equation library and the constraint programming solver with the problem specific branching heuristic are described in detail.

4.2. Hierarchical Performance Equation Library (HPEL)[1]

The hierarchical performance equation library uses the functional block description of op-amps to store the equations describing the op-amp behavior topology independent. It distinguishes between two main groups of equations: the basic model and the op-amp performance model.

The *basic model* describes the current and voltage flow in the circuit. It contains information obtained from the circuit netlist and an analysis of its devices. It comprises the variables of the circuit, Kirchoff's Current and Voltage Law and models for the devices. The variables can be reduced by using information from higher functional block levels, however the major set-up is based on the first hierarchy level.

The *op-amp performance model* describes the ac-behavior, dc-behavior, and transient behavior of the op-amp. It contains information gained from the hierarchical composition of

[1]A similar version of this section was published in [59], reprinted with permission from "Inga Abel, Maximilian Neuner, Helmut Graeb, A Hierarchical Performance Equation Library for Basic Op-Amp Design, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems" © 2021 IEEE.

Table 4.1.: Hierarchical performance equation library

	Symmetry constraints	Functional block behavioral constraints	Intermediate performance equations	Op-amp performance equations
HL 1: Devices			• Saturation drain-source voltage • Net capacitance	• Area • Quiescent power
HL 2: Structures	• Voltage and current bias	• Current mirror behavior		
HL 3: Amplification stage subblocks	• Load • Non-inverting transconductor	• Complementary transconductor and stage bias	• Transconductance • Output conductance	
HL 4: Op-amp subblocks	• Inverting stages	• Output voltage offset	• Stage output resistance • Stage open-loop gain • Stage non-dominant poles • Stage zeros	• Common-mode input voltage • Output voltage • Common-mode rejection ratio • Unity-gain bandwidth
HL 5: Op-amps			• Dominant pole • Positive zero	• Open-loop gain • Slew rate • Phase margin

functional blocks. It comprises symmetry constraints, functional block constraints, intermediate performance equations and op-amp performance equations. An overview of the op-amp performance model part of the hierarchical equation library is given in Table 4.1. For each hierarchy level, the most important equations or constraints are given. The ordering from left to right represents an abstraction from constraints to performance and it corresponds to the functional abstraction from top to bottom through the hierarchy levels. Equations based on functional blocks of low hierarchy levels highly depend on the transistor structure, e.g., the equations to describe the output conductance of a functional block. To set up the equations of the open-loop gain, the transistor structure of op-amp is ignored. This hierarchical structuring allows us to generalize the op-amp equations such that we obtain an automatic set-up of the design equations independent of the topology (Sec. 4.3).

In the following sections, variables and equations of both model types are described in detail.

4.2.1. Variables

The variables of the equation-based topology description can be divided into two groups: device specific variables and op-amp performance variables.

Device specific variables: Depending on the device type, a set of variables is automatically derived. For a transistor t_k, this set is:

$$\mathbf{t}_k^T = \{w_k, l_k, gm_k, gd_k, i_{DS,k}, v_{GS,k}, v_{DS,k}\} \tag{4.1}$$

w_k, l_k are the width and length of the transistor, gm_k, gd_k its transconductance and output conductance, $i_{DS,k}$ its drain-source current and $v_{GS,k}, v_{DS,k}$ its gate-source and drain-source voltage.

Op-amp performance variables: The set of characteristic performance features whose equations are automatically set up based on the HPEL is:

$$\mathbf{z}^T = \{z_D, z_{QP}, z_{v_{cm,min/max}}, z_{v_{out,min/max}}, z_{f_{GBW}}, z_{SR}, z_{A_{D0}}, z_{CMRR}, z_{PM}\} \tag{4.2}$$

z_D describes the gate-area of the circuit, z_{QP} its quiescent power, $z_{v_{cm,min/max}}$ is the minimal, respective maximal common-mode input voltage, $z_{v_{out,min/max}}$ the minimal/maximal output voltage of the op-amp, $z_{A_{D0}}$ is its open-loop gain. $z_{f_{GBW}}$ is the unity-gain bandwidth, z_{SR} the slew rate, z_{CMRR} the common-mode rejection ratio, z_{PM} is the phase margin.

The performance features describe the characteristic op-amp behavior. Additional to them, intermediate performance variables, e.g., output resistance R_{out,a_j} of a stage a_j or poles and zeros of an op-amp $\mathbf{z}_P$ exist.

4.2.2. Kirchoff's Current and Voltage Law

By automatically analyzing the graph description of the netlist, Kirchoff's Current Law (KCL) is set up for every node $l \in \mathcal{N}$ in the circuit:

$$\forall_{l \in \mathcal{N}} \sum_k i_{DS_k} = 0, \tag{4.3}$$

Kirchoff's Voltage Law is expressed efficiently by the voltage potentials of the circuit nodes as in circuit simulation. The voltage variables are therefore all n_N node voltages $\mathbf{v}_N$:

$$\begin{aligned} &\mathbf{v}_N^T = [v_{N,1}, v_{N,2}, ..., v_{N,n_N}], v_{N,k} \in \mathbb{R}, k = 1, 2, ..., n_N \\ &[\mathbf{v}_{GS}^T \mathbf{v}_{DS}^T]^T = \mathbf{A} \cdot \mathbf{v}_N, \text{ with } \mathbf{A} \text{ as nodal incidence matrix.} \end{aligned} \tag{4.4}$$

4.2.3. Transistor Behavior Model

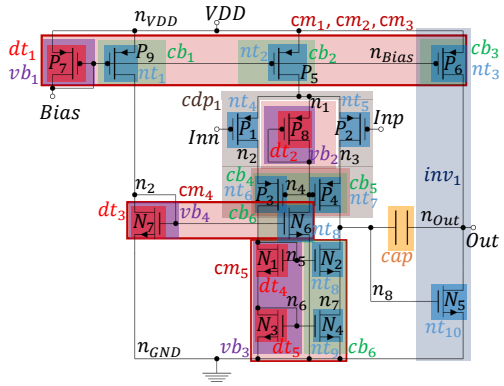
For every device in the circuit, a model is needed which describes its behavior depending on its variables. Therefore, for every device type on HL 1, a behavior model is stored in the equation library. For transistors, this is the Shichman-Hodges model [71]. It is the simplest transistor model and defines three operating regions for a transistor, off, linear, and saturation. Analytical equations exist for all operating regions. The drain-source current of an NMOS transistor t_k in saturation is for instance described by:

$$i_{DS,k} = \frac{\mu_k C_{ox,k}}{2} \frac{w_k}{l_k} (v_{GS,k} - v_{th,k})^2 (1 + \lambda_k v_{DS,k}) \tag{4.5}$$

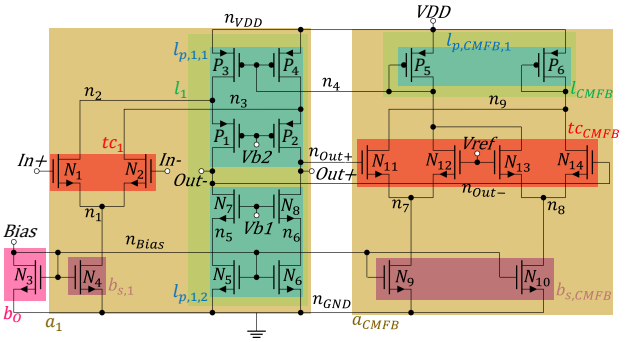
Process parameters, e.g., threshold voltage v_{th}, are specified by the underlying process technology. Equations for other operation regions are implemented analogously.

Constraints for the transistor voltages guarantee that the transistor operates in the specified region. For saturation, these are:

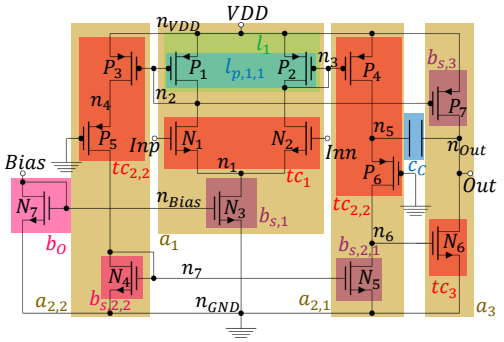
$$\begin{aligned} v_{GS,k} - v_{th,k} &\geq 0 \\ v_{GS,k} - v_{th,k} &< v_{DS,k} \end{aligned} \tag{4.6}$$



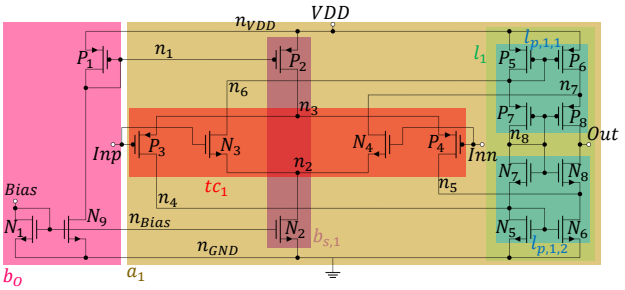
(a) Telescopic op-amp [colored background: functional blocks of HL 1 - 2]



(b) Folded-cascode op-amp with Common-Mode Feed-Back (CMFB) [colored background: functional blocks of HL 3 - 4]



(c) Symmetrical op-amp with high Power-Supply Rejection Ratio (PSRR) [48] [colored background: functional blocks of HL 3 - 4]



(d) Complementary op-amp [colored background: functional blocks of HL 3 - 4]

Figure 4.2.: Different op-amp topologies

The transconductance and output conductances gm_k, gd_k of a transistor are calculated by the differentiation of the drain-source current with respect to the transistor voltages. For the saturation region, following equations are obtained:

$$gm_k = \frac{\partial i_{DS,k}}{\partial v_{GS,k}} = \sqrt{2\mu_k C_{ox,k} \frac{W_k}{L_k} i_{DS,k}} \tag{4.7}$$

$$gd_k = \frac{\partial i_{DS,k}}{\partial v_{DS,k}} = \lambda_k \cdot i_{DS,k} \tag{4.8}$$

The saturation region can be further divided into weak, moderate and strong inversion. An overview how the three inversion regions are integrated into equation-based modeling is given in [60].

4.2.4. Symmetry Constraints

Symmetry constraints are crucial in analog circuits to minimize mismatch, e.g., due to channel length modulation or local manufacturing variations. Symmetry constraints are derived

for structures (HL 2), subblocks of amplification stages (HL 3) and op-amp subblocks (HL 4). They reduce the number of variables of the performance model.

Hierarchy Level 2: Structures

For transistors in a voltage or current bias, we define that two transistors t_i, t_j connected at their gates $t_i.g, t_j.g$ must have equal lengths l_{t_i}, l_{t_j}

$$\forall_{t_i,t_j \in (T_{vb,\Phi} \cup T_{cb,\Phi})} t_i.g \leftrightarrow t_j.g \Rightarrow l_{t_i} = l_{t_j} \tag{4.9}$$

$T_{vb,\Phi}$ is the set of transistors of doping Φ being part of the voltage biases in the circuit. $T_{cb,\Phi}$ is the set of transistors of the same doping Φ being part of current biases.

Hierarchy Level 3: Amplification Stage Subblocks

The dc-current flow must be symmetric in the subblocks of a non-inverting stage. We therefore define that the transconductor of the non-inverting stage $tc_{ninv,k}$ and its load l_k must have symmetrical geometries:

$$\forall_{tc_{ninv,k} \in \mathcal{M}_{tc_{ninv}}} \{t_{k,i,\Phi}, t_{k,j,\Phi}\} = tc_{ninv,k} \ \wedge \ l_{t_{k,i}} = l_{t_{k,j}} \ \wedge \ w_{t_{k,i}} = w_{t_{k,j}} \tag{4.10}$$

$$\forall_{t_i,t_j \in (T_l)} t_i.g \leftrightarrow t_j.g \Rightarrow (l_{t_i} = l_{t_j} \wedge w_{t_i} = w_{t_j}) \tag{4.11}$$

$\mathcal{M}_{tc_{ninv}}$ is the set of non-inverting transconductors in the op-amp and T_l the set of transistors forming the load.

Hierarchy Level 4: Op-Amp Subblocks

Symmetrical op-amps and fully-differential two-stage op-amps have two second stages. They must be symmetrical:

$$\forall_{t_m \in a_{2,1}, t_n \in a_{2,2}} \left[(t_m.pos = t_n.pos) \rightarrow (w_{t_m} = w_{t_n} \wedge l_{t_m} = l_{t_n})\right] \tag{4.12}$$

$t_k.pos$ gives the position of a transistor, e.g., n-type transistor that is connected to the ground net. The transistors on equal positions should have equal geometries. In the symmetrical op-amp with high PSRR in Fig. 4.2c, the transistors N_4, N_5 have equal positions and therefore should have the same sizes. The other transistor pairs are P_5, P_6 and P_3, P_4.

4.2.5. Functional Block Behavior Constraints

Behavior constraints on a functional block are constraints on its transistor variables required to ensure the proper functionality of the block. Behavior constraints are derived for structures (HL 2), amplification stage subblocks (HL 3) and op-amp subblocks (HL 4).

Hierarchy Level 2: Structures

Behavior constraints for specific types of current mirrors are instantiated on this level. An example is the cascode current mirror (e.g., Fig. 4.2a, $N_1 - N_4$). In this type of current mirror, the voltage potentials of the inner nets, e.g., Fig. 4.2a, n_6, n_7, must be equal to suppress the effect of the channel length modulation. To obtain equal voltages, the ratio of the widths of the transistors in the current mirror must be restricted:

$$\frac{w_{ccm,vb,d}}{w_{ccm,cb,d}} = \frac{w_{cm,vb,s}}{w_{cm,cb,s}} \tag{4.13}$$

$w_{ccm,vb,d}, w_{ccm,vb,s}$ are the drain and the source transistor of the voltage bias in the cascode current mirror. $w_{ccm,cb,d}, w_{ccm,cb,s}$ are the drain and the source transistor of the current bias in the cascode current mirror. Please note that the transistor length is already restricted by (4.9).

Hierarchy Level 3: Amplification Stage Subblocks

A behavior constraint on the amplification stage subblock level exists for the complementary transconductor tc_c. The transconductance of the transistors in the n-doped differential pair $gm_{dp,n,i}|_{i=1,2}$ and p-doped differential pair $gm_{dp,p,j}|_{j=1,2}$ of tc_c must be equal.

$$gm_{dp,n,i}|_{i=1,2} = gm_{dp,p,j}|_{j=1,2} \tag{4.14}$$

Also the currents of the differential pairs generated with the n-doped and p-doped transistors in the stage bias $b_{s,c}$ must be equal:

$$|i_{DS,b_{s,c,n}}| = |i_{DS,b_{s,c,p,}}| \tag{4.15}$$

Hierarchy Level 4: Op-Amp Subblocks

A constraint on the op-amp subblock level is the output voltage offset constraint for two stage op-amps. To suppress an offset voltage on the output voltage by equal input voltage, the voltage potentials at the first stage output nets $n_{a_1.out_1}, n_{a_1.out_2}$, e.g., Fig. 4.2a, $n_{a_1.out_1} = n_5, n_{a_1.out_2} = n_8$ must be equal.

$$a_2 \in \mathcal{M} \Rightarrow v_{n_{a_1.out_1}} = v_{n_{a_1.out_2}} \tag{4.16}$$

$\mathcal{M}$ is the set of functional blocks of an op-amp topology.

4.2.6. Intermediate Performance Equations

Hierarchy levels 1, 3-5 are considered to establish all intermediate performance equations. Please note that these equations are only instantiated for a functional block if they are required by an op-amp performance equation.

Hierarchy Level 1: Devices

The saturation drain-source voltage of a transistor and the net capacitance of a net in the circuit are equations generated based on the device information.

Saturation Drain-Source Voltage The saturation drain-source voltage is the voltage at least needed to keep a transistor in saturation. According to (4.6), this is:

$$v_{DS,sat,i} = \begin{cases} v_{GS,i} - v_{th,i}, & t_i.type = nt \\ v_{GS,i}, & t_i.type = dt \end{cases} \tag{4.17}$$

considering that for a diode transistor dt_k, $v_{GS,k} = v_{DS,k}$.

Net Capacitance The capacitance C_{n_i} of a net n_i depends on the device pins P_{n_i} connected to n_i:

$$C_{n_i} = \sum_{p_j \in P_{n_i}} C_{p_j} \tag{4.18}$$

C_{p_j} is the capacitance arising by the pin p_j. The corresponding equations to calculate C_{p_j} are given in [80] and are shown in Appendix A.1.

Hierarchy Level 3: Amplification Stage Subblocks

Transconductances and output conductances of the functional blocks are important properties to be described on this level.

Transconductance The transconductance of a transconductor is defined by one of the transistors $t_{tc,i,in}$ whose gate is connected to the input signal of the stage.

$$gin_{tc,i} = gm_{t_{tc,i,in}} \tag{4.19}$$

In a non-inverting stage, $t_{tc,i,in}$ is one of the transistors of the differential pair. Due to symmetry, the transconductance of both transistors is equal. In an inverting stage, $t_{tc,i,in}$ is the transistor whose gate is connected to the output of the previous stage. In op-amps with two second stages, the transconductance of only one of the two stages must be calculated due to symmetry.

For the calculation of the transconductance of the complementary transconductor (Fig. 4.2d), the transconductances of the transistors in the PMOS differential pair and the NMOS differential pair must be considered:

$$gin_{tc_c} = gm_{t_{tc,in,n}} + gm_{t_{tc,in,p}} \tag{4.20}$$

Output Conductance The computation of $gout_i$ for a functional block m_i depends on its inner structure. It is distinguished between functional blocks consisting of one- and two-transistor stacks. A transistor stack is defined as a sequence of transistors having a drain-source connection (Chapter 3). N_1, N_3 in Fig. 4.2a is an example of a two-transistor stack.

$$gout_i = \begin{cases} gd_{t_{i,out}}, & \{t_{i,out}\} = ts_i \subseteq m_i \\ \frac{gd_{t_{i,out}} gd_{t_{i,supply}}}{gm_{t_{i,out}}}, & \{t_{i,out}, t_{i,supply}\} = ts_i \subseteq m_i \end{cases} \tag{4.21}$$

If the functional block m_i consists of one-transistor stacks, only the output conductance of the transistor in m_i connected to the stage output is relevant for the calculation. If m_i consists of two-transistor stacks, the transistor connected to the stage output and the transistor connected to the supply voltage rail are relevant. The type of functional block as load, transconductor and stage bias is irrelevant. All three types consist of one or two transistor stacks. If the functional block consists of two transistor stacks, the stacks are symmetrical. In non-symmetrical load parts (Chapter 3), the output-connected transistor stack is only relevant for calculations. Hence, in Fig. 4.2c, $gout_{tc_1} = gd_{N_1} = gd_{N_2}$ and $gout_{tc_{2,2}} = \frac{gd_{P_5} gd_{P_3}}{gm_{P_5}}$.

Further differentiation must be made for load parts including a gate-connected couple being part of a cascode or folded-cascode differential pair (Chapter 3):

$$gout_i = \begin{cases} \frac{gd_{t_{i,out}} gd_{t_{tc_1,1}}}{gm_{t_{i,out}}}, & \{t_{i,out}\} = ts_i \subseteq m_i \wedge \{t_{tc_1,1}, t_{i,out}\} \subset cdp_k \\ \frac{gd_{t_{i,out}} (gd_{t_{i,supply}} + gd_{t_{tc_1,1}})}{gm_{t_{i,out}}}, & \{t_{i,out}, t_{i,supply}\} = ts_i \subseteq m_i \wedge \{t_{tc_1,1}, t_{i,out}\} \subset fcdp_k \end{cases} \tag{4.22}$$

The output conductance of one of the transistors of the differential pair must be included in these calculations. Thus, for the load part formed by P_3, P_4 in Fig. 4.2a $gout_{l_{p,1,1}} = \frac{gd_{P_4} gd_{P_2}}{gm_{P_4}}$ and $gout_{l_{p,1,1}}$ in Fig. 4.2b is $gout_{l_{p,1,1}} = \frac{gd_{P_2}(gd_{P_4} + gd_{N_2})}{gm_{P_2}}$.

In symmetrical op-amps and Common-Mode FeedBack (CMFB) stages, the output conductance $gout_i$ of the load part is the transconductance of one of the transistors connected with its gate to the output of the stage. The load part consists only of voltage biases (Chapter 3):

$$gout_i = gm_{t_{g.out}},\ m_i.type = l_p \wedge m_i = \{vb_{i,1}, vb_{i,2}\} \tag{4.23}$$

Hence, in Fig. 4.2c, $gout_{l_{p,1,1}} = gm_{P_1} = gm_{P_2}$

Hierarchy Level 4: Op-Amp Subblocks

The output resistance, the open-loop gain and the non-dominant poles and zeros of an amplification stage are calculated on this level.

Stage Output Resistance The output resistance of an amplification stage is described by the output conductances of the k functional blocks of the stage, e.g., stage biases, load parts and transconductors, connected to the stage output net.

$$R_{out,i} = \frac{1}{\sum_{j=1}^{k} gout_i} \tag{4.24}$$

For the symmetrical op-amp with high PSRR (Fig. 4.2c), the output resistance of the first stage R_{out,a_1} is for example calculated by $gout_{tc_1}$ and $gout_{l_{p,1,1}}$. For the folded-cascode first stage in Fig. 4.2b, the output resistance is $R_{out,a_1} = \frac{1}{gout_{l_{p,1,1}} + gout_{l_{p,1,2}}}$.

Stage Open-Loop Gain The open-loop gain of a stage $A_{D0,i}$ is calculated by the transconductance $gin_{tc,i}$ of its transconductors (4.19) and its output resistance $R_{out,i}$:

$$A_{D0,i} = gin_{tc,i} \cdot R_{out,i} \tag{4.25}$$

Stage Non-Dominant Poles Non-dominant poles arise for every stage in the op-amp. They must be calculated for every transistor t_k on the signal path from input to output. The pole of the transistor t_k is calculated by:

$$f_{ndp,t_k} = \frac{gm_{t_k}}{2\pi C_{n_j}} \tag{4.26}$$

C_{n_j} is the net capacitance of the net the signal passes by before encountering t_k. It is calculated by (4.18) and contains the parasitics emerging from the transistor pins as well as the capacitance of the capacitors connected to the net.

If a compensation capacitor is connected between the input and the output of a stage, the equation of the non-dominant pole at the input transistor of the stage changes to:

$$f_{ndp,inv,C_c} = \frac{gin_{tc_{inv}}}{2\pi(C_{n_{out}} + \frac{C_{n_{tc,inv,in,g}} \cdot C_{n_{out}}}{C_C} + C_{n_{tc,inv,in,g}})} \tag{4.27}$$

$gin_{tc_{inv}}$ is the transconductance of the stage. $C_{n_{tc,inv,in,g}}$ is the capacitance of the gate net carrying the input signal of the stage. $C_{n_{out}}$ is the capacitance of the output net of the stage and C_C the capacitance value of the compensation capacitor.

Appendix A.2 lists non-dominant poles additional to (4.27), which have a noticeable influence on the phase margin.

Stage Zeros Zeros are evoked by non-dominant poles if they are a mirror pole, i.e., only half of the signal is influenced by it. They are set to occur at twice of the frequency of the mirror pole $f_{ndp,mir}$:

$$f_{z,mir} = 2 \cdot f_{ndp,mir} \tag{4.28}$$

The appendix A.3 lists zeros in op-amps relevant for phase margin calculation.

Hierarchy Level 5: Op-Amp

The complete composition of the op-amp must be considered to calculate the dominant pole and the positive zero.

Dominant Pole The dominant pole is the pole at the smallest frequency in an op-amp. It occurs mostly at the output net of the first stage and is calculated by:

$$f_{dp} = \frac{1}{2\pi C_{n_{out}} gin_{tc_2} \Pi_{i=1}^{2} R_{out,i}} \tag{4.29}$$

$C_{n_{out}}$ is the capacitance at the output net of the first stage, gin_{tc_2} the transconductance of the second stage transconductor and $R_{out,j}$ the output resistance of the amplification stages. For single-stage op-amps, gin_{tc_2} and $R_{out,2}$ are set to one.

In symmetrical op-amps, the dominant pole occurs at the output net of the second stage. The equation changes to:

$$f_{dp} = \frac{1}{2\pi C_{n_{out}} gm_{tc_3} \Pi_{i=2}^{3} R_{out,i}} \tag{4.30}$$

where $C_{n_{out}}$ is the capacitance at the output net of the second stage. If no third stage is part of the symmetrical op-amp, gm_{tc_3} and $R_{out,3}$ are set to one.

Positive Zero In op-amps with compensation capacitor c_C, a positive zero exists. It is calculated by:

$$f_{pz} = \frac{1}{2\pi C_C (\frac{1}{gin_{tc_{inv,k}}} - \frac{1}{gd_{R_C}})} \tag{4.31}$$

$gin_{tc_{inv,k}}$ is the transconductance of the inverting transconductor connected by c_C to a previous stage. If a compensation resistor R_C is part of the circuit, gd_{R_C} is the output conductance of the transistor emulating the compensation resistor, otherwise $gd_{R_C} = 1$.

4.2.7. Op-Amp Performance Equations

Analogous to the equations and constraints before, the performance features of an op-amp are ordered hierarchically. Some performance equations only need the device level information as input (HL 1). Others are based on op-amp subblocks (HL 4) or on the whole op-amp (HL 5).

Hierarchy Level 1: Devices

The area and quiescent power of the op-amp are calculated based on device level information.

Area An estimation of the area of the circuit is calculated by the gate-areas of all k transistors:

$$z_D = \sum_{i=1}^{k} w_i \cdot l_i \tag{4.32}$$

Quiescent Power The quiescent power z_{QP} is the product of the positive supply voltage v_{VDD} subtracted by negative voltage v_{VSS} with the sum of the n currents flowing into the positive supply voltage net n_{VDD}. If the bias current of the circuit i_{Bias} is applied to an NMOS transistor, it must be added to the currents flowing into n_{VDD}.

$$z_{QP} = (v_{VDD} - v_{VSS}) \cdot \begin{cases} \sum_{j=1}^{n} |i_j|, & t_{Bias}.\Phi = p \\ \sum_{j=1}^{n} |i_j| + i_{bias}, & t_{Bias}.\Phi = n \end{cases} \tag{4.33}$$

Hierarchy Level 4: Op-Amp Subblocks

Performance features determined by one amplification stage are formulated on this level. These are common-mode input voltage, output voltage, common-mode rejection ratio (CMRR) and unity-gain bandwidth.

Common-Mode Input Voltage The common-mode input voltage describes the range in which the input voltage can vary without changing the behavior of the op-amp. We can specify a maximum $z_{v_{cm,max}}$ and a minimum $z_{v_{cm,min}}$ common-mode input voltage. The two voltage loops which describe $z_{v_{cm,max}}, z_{v_{cm,min}}$ are either over the load of the first stage l_1 or over its stage bias $b_{s,1}$. Therefore, we can define the two limiting values by $v_{cm,l_1}, v_{cm,b_{s,1}}$. $z_{v_{cm,max}}, z_{v_{cm,min}}$ are defined depending which of $v_{supply,b_{s,1}}, v_{supply,l_1}$ equals v_{VDD}, v_{VSS}.

$$\begin{aligned} v_{supply,b_{s,1}} = v_{VDD} \wedge v_{supply,l_1} = v_{VSS} \Rightarrow z_{v_{cm,max}} = v_{cm,b_{s,1}} \wedge z_{v_{cm,min}} = v_{cm,l_1} \\ v_{supply,b_{s,1}} = v_{VSS} \wedge v_{supply,l_1} = v_{VDD} \Rightarrow z_{v_{cm,max}} = v_{cm,l_1} \wedge z_{v_{cm,min}} = v_{cm,b_{s,1}} \end{aligned} \tag{4.34}$$

For loads connected to both supply voltage rails, e.g., Fig. 4.2b, the supply voltage rail opposite to $v_{supply,b_{s,1}}$ is considered. If the transistors in the paths are in saturation and in strong inversion, $v_{cm,b_{s,1}}$ and $v_{cm,l_{p,1,1}}$ are defined by the minimum/maximum voltage which is allowed when keeping all transistors in saturation. For a single transistor, this voltage is defined by the minimum saturation voltage (4.17). Hence, $v_{cm,b_{s,1}}$ is defined as:

$$v_{cm,b_{s,1}} := v_{supply,b_{s,1}} + v_{GS,tc_1} + \sum_{i=1}^{|b_1|} v_{DS,sat,i} \tag{4.35}$$

Determining v_{cm,l_1} is more complex, as the structure of the load highly varies (Chapter 3). Two relevant voltage paths e_1, e_2 exist having the smallest possible number of voltage drops. Each path starts from one of the outputs of the first stage transconductor tc_1 going to the supply-voltage rail of the load n_{supply,l_1}. In the folded-cascode op-amp with CMFB (Fig. 4.2b), $e_1 = \{P_3\}, e_2 = \{P_4\}$. In the telescopic op-amp (Fig. 4.2a),

$e_1 = \{P_3, N_1, N_3\}, e_2 = \{P_4, N_2, N_4\}$. For the calculation of v_{cm,l_1}, the path e_k is chosen with the most diode transistors. For diode transistors, $v_{DS,sat} = v_{GS}$. Thus, they have a much higher impact on the input voltage range as normal transistors. If the transistors in the paths are in saturation and in strong inversion, v_{cm,l_1} is defined by:

$$v_{cm,l_1} := v_{supply,l_1} + v_{th,tc_1} + \sum_{m=1}^{|e|} \begin{cases} -(v_{DS,sat,m}), & t_m \subset gcc_m \\ v_{DS,sat,m}, & else \end{cases} \tag{4.36}$$

v_{th,tc_1} is the threshold voltage of a transistor of the transconductor of the first stage tc_1. If a transistor of a gate-connected couple gcc_k is in the path e, e.g., P_3, P_4 in Fig. 4.2a, it introduces a negative value for $v_{DS,sat,k}$. It has a different substrate doping than the other transistors in the load relevant for e.

Output Voltage The output voltage swing is described by the last stage of an op-amp. A maximum value $z_{v_{out,max}}$ and a minimum value $z_{v_{out,min}}$ are defined by the shortest paths from the output of the op-amp to the supply-rails e_{VDD}, e_{VSS}. The paths contain the transistors being part of transistor stacks connecting the supply-voltage rails to the output. If the transistors are supposed to be in saturation and in strong inversion, the corresponding equations are:

$$z_{v_{out,max}} = v_{VDD} + \sum_{i=1}^{|e_{VDD}|} v_{DS,sat,i} \tag{4.37}$$

$$z_{v_{out,min}} = v_{VSS} + \sum_{i=1}^{|e_{VSS}|} v_{DS,sat,i} \tag{4.38}$$

with $v_{DS,sat,i}$ described by (4.17).

Common-Mode Rejection Ratio For non-fully-differential op-amp topologies, an analytical equation can be derived that gives a good approximation of the static systematic common-mode rejection ratio (CMRR_s). For all op-amps but symmetrical op-amps, the CMRR_s only depends on the structure of the first stage.

$$z_{CMRR} = 2A_{D0,1} \cdot \frac{gm_{l_1,g.out}}{gout_{b_{s,1}}} \tag{4.39}$$

$A_{DO,1}$ is the open-loop gain of the first stage of the op-amp (4.25). $gout_{b_{s,1}}$ the output conductance of the first stage bias calculated according to (4.21). $gm_{l_1,g.out}$ is the transconductance of the load transistor connected with its gate to one of the output nets of the first stage. If the gates of two load transistors are connected to the output of the first stage, any of these two can be chosen. They have equal gm-values, as the load of an op-amp is symmetric. If no load transistor's gate is connected to the output of the first stage the equation to calculate the CMRR_s changes:

$$z_{CMRR} = 2 \cdot \frac{gin_{tc_1}}{gout_{b_{s,1}}} \tag{4.40}$$

gin_{tc_1} is the transconductance of the transconductor of the first stage calculated according to (4.19).

The CMRR of a symmetrical op-amp is also defined by the open-loop gain of the second stage $A_{D0,2}$:

$$z_{CMRR_{sym}} = 2A_{D0,1} \cdot A_{D0,2} \cdot \frac{gm_{l_1,g.out}}{gout_{b_{s,1}}} \tag{4.41}$$

For fully-differential op-amps, the CMRR$_s$ also depends on the common-mode feedback circuit and is not discussed here. In complementary op-amps, the two stage bias types of the first stage, PMOS and NMOS, must be considered.

Unity-Gain Bandwidth The unity-gain bandwidth $z_{f_{GBW}}$ is calculated by the transconductor of the first stage tc_1 and the capacitance of the first stage output net $C_{n_{a_1,out}}$:

$$z_{f_{GBW}} = \frac{gin_{tc_1}}{2\pi C_{n_{a_1,out}}} \tag{4.42}$$

The equation for $z_{f_{GBW}}$ differs slightly for symmetrical op-amps, as the second stage impacts the unity-gain bandwidth:

$$z_{f_{GBW}} = \frac{A_{D0,1} \cdot gin_{tc_2}}{2\pi C_{n_{a_2,out}}} \tag{4.43}$$

$A_{D0,1}$ is the open-loop gain of the first stage, $gin_{tc_2,}$ the transconductance of the second stage transconductor, and $C_{n_{a_2,out}}$ the capacitance of the second stage output net connected to a capacitor.

Hierarchy Level 5: Op-Amp

The overall op-amp structure is considered for the calculation of open-loop gain, the slew rate and the phase margin.

Open-Loop Gain The open-loop gain of an op-amp is the product of the open-loop gains of its n stages:

$$z_{A_{D0}} = \prod_{i=1}^{n} A_{D0,i} \tag{4.44}$$

Slew Rate The slew rate z_{SR} of a circuit is calculated by the bias currents of the n stages and the capacitances of the output nets of the stages,

$$z_{SR} = min\{\frac{|i_{DS,b_{s,1}}|}{C_{n_{out,1}}}, .., \frac{|i_{DS,b_{s,n}}|}{C_{n_{out,n}}}\} \tag{4.45}$$

where $i_{DS,b_{s,k}}$ is the drain-source current of a transistor part of the stage bias $b_{s,k}$ of the stage k. $C_{n_{out,k}}$ is the capacitance of the stage output net calculated by (4.18). For symmetrical op-amps, the first stage output net does not have to be considered. However, if one of the input transistors of the first stage is shut down, the bias current of the first stage is amplified and mirrored by the current mirror forming the first stage load and the second stage transconductor, e.g., Fig. 4.2c P_2, P_4. Therefore, twice the bias current of the second stage must be considered during slew rate calculations. In a folded-cascode op-amp, the current $i_{DS,l_{B,GCC}}$ of the two transistors biasing the gate-connected couple, e.g., Fig. 4.2b P_3, P_4, must be considered during slew rate calculation. The smallest current of $i_{DS,l_{B,GCC}}$, $i_{DS,b_{s,1}}$ restricts the slew rate.

Phase Margin The phase margin z_{PM} is calculated by the non-dominant poles and zeros of the circuit:

$$z_{PM} = \frac{\pi}{2} - \sum_{i=1}^{m} \text{atan}(\frac{f_{GBW}}{f_{ndp_i}}) + \sum_{j=1}^{n} \text{atan}(\frac{f_{GBW}}{f_{z_j}}) \tag{4.46}$$

f_{GBW} is the unity-gain bandwidth of the circuit. A positive zero has a negative influence on the phase margin, like non-dominant poles. The non-dominant poles and zeros must be at least an order of magnitude larger than the dominant pole.

$$\forall_{f_i \in (F_{ndp} \cup F_z)} \frac{f_i}{f_{dp}} > 10 \tag{4.47}$$

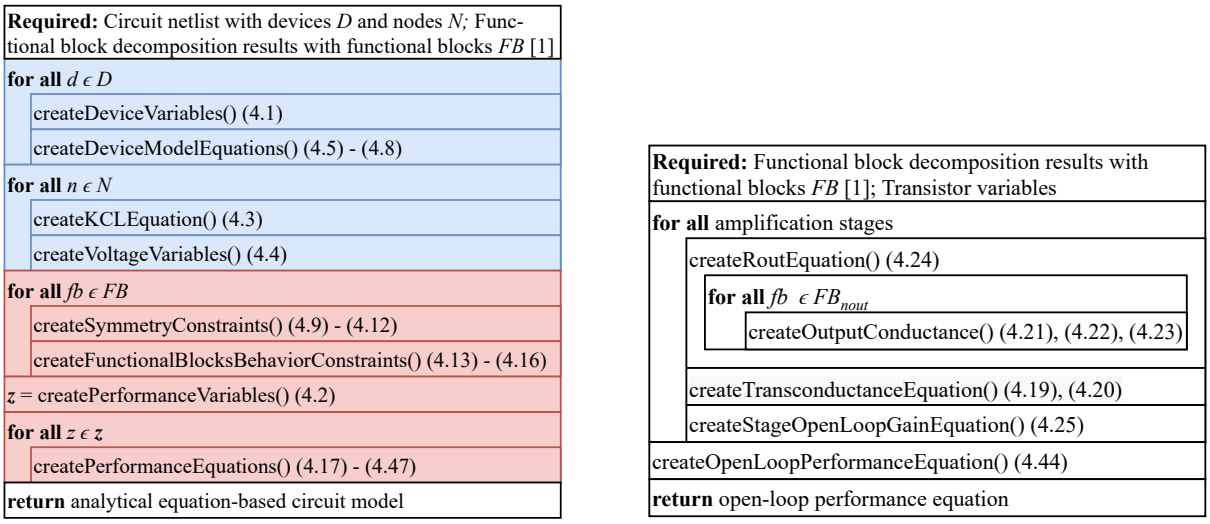
4.3. Automatic Instantiation of the Equation-Based Circuit Model Based on the HPEL[2]

Fig. 4.3 shows the automatic synthesis of the equation-based circuit model for a given op-amp topology. The input of the algorithm are the circuit netlist and the results of the functional block decomposition method in Chapter 3, which automatically identifies all functional blocks in a circuit netlist. The basic circuit model and the circuit performance model are automatically instantiated based on this input.

The algorithm iterates over the devices and nodes in the circuit to create the basic model. The corresponding variables and equations are automatically instantiated. This is similar to a circuit simulation tool.

Symmetry constraints, functional block behavior constraints and performance equations are automatically created to form the op-amp performance model. The symmetry and functional

[2]A similar version of this section was published in [59], reprinted with permission from "Inga Abel, Maximilian Neuner, Helmut Graeb, A Hierarchical Performance Equation Library for Basic Op-Amp Design, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems" © 2021 IEEE.



(a) Main algorithm

(b) Performance equation creation on the example of the open loop gain

Figure 4.3.: Automatic instantiation of an equation-based circuit model for a given topology

block behavior constraints are created by iterating over all recognized functional blocks, instantiating the corresponding constraints by selecting the corresponding variables of the basic circuit model. This is similar to the method in [52], which creates constraints for basic transistor pairs.

The performance equations are set up for every performance variable in (4.2). Every equation stated on a high level of abstraction in Sec. 4.2.7 is broken down into the circuit variables using the intermediate performance equations. Fig. 4.3b illustrates this procedure with the open-loop gain. To instantiate the open-loop gain performance equation (4.44), the open-loop gain equations of the individual amplification stages must be created. These equations take the output resistances and the transconductances of the stages as input (4.25). The transconductance of a stage in turn takes the circuit variables as input (4.19). For the equation of the output resistance, the equations of the output conductances of all functional blocks on HL 3 $FB_{n_{out}}$ connected to the output net n_{out} must be created. The equations of the output conductances have the circuit variables as input (4.21)-(4.23). Thus, an overall open-loop gain equation is automatically instantiated with the circuit variables as input.

Analogously to Fig. 4.3b, the performance equations for every supported performance feature in (4.2) are automatically instantiated for a given topology linking the abstract performance equations in Sec. 4.2.7 to the circuit variables using the intermediate performance equations (Sec. 4.2.6). Many intermediate performance equations are part of several different op-amp performance equations. The transconductance of the first stage is for example part of the open-loop gain equation as well as part of the equation for the unity-gain bandwidth. The equations are stored topology independent but customized by the algorithms. In contrast to the presented approach, the state of the art is limited to individual topologies and their specific equation sets.

4.4. Constraint Programming - A Backtracking Search-Based Solver Adoptable to MINLP's[3]

The automatically created circuit model (Sec. 4.3) is fed into a constraint programming solver to solve it for the device sizes. Constraint Programming is especially suitable for the combinatorial character of analog sizing due to the manufacturing-induced discrete value range of transistor geometries and allows all function types as, e.g., trigonometrical, polynomial. No further approximations have to be made to the performance equations. In the following sections, an overview of Constraint Programming is given.

Constraint Programming [81–86] is a programming paradigm which allows to model a mathematical problem with a set of variables $\mathbf{z}$, a domain D_i for each variable $z_i \in \mathbf{z}$ and a set of constraints C:

$$\mathbf{z} = \{z_1, z_2, ..., z_n\},\ z_i \in D_i,\ i \in [1, .., n] \tag{4.48}$$

$$C = E \cup I = \{c_1, c_2, ..., c_m\} \tag{4.49}$$

The domains D_i contain either boolean, integer or float values and are either finite or infinite. A constraint $c_j \in E$ is formulated as equality constraint, i.e., $c_j(\mathbf{z}) = 0$, a constraint $c_k \in I$ as inequality constraint, i.e., $c_j(\mathbf{z}) \bigcirc 0$, with $\bigcirc \in \{>,\ <,\ \geq,\ \leq,\ !=,\ ...\}$. Without loss of generality, inequality constraints are formulated with "$\geq$" in the following. The constraints $c \in C$ form a constraint satisfaction problem CSP:

$$CSP(\mathbf{z}) \Leftrightarrow 1, IFF\ \forall_{j \in E}\ c_j(\mathbf{z}) = 0 \wedge \forall_{k \in I}\ c_k(\mathbf{z}) \geq 0, \tag{4.50}$$

which must be fulfilled. The initial search space S of a CSP is defined as:

$$S := D_1 \times D_2 \times ... \times D_n \tag{4.51}$$

A constraint solver computes all solutions, i.e., value assignments of $\mathbf{z}$, which fulfill the CSP.

If an additional vector of target functions $\mathbf{t}(\mathbf{z})$ is given, a constraint optimization problem (COP) can be formulated as

$$\max\ \ \mathbf{t}(\mathbf{z})\ \ s.t.\ \ CSP(\mathbf{z}) \Leftrightarrow 1. \tag{4.52}$$

Solving (4.52) yields the maximum solution of $\mathbf{t}(\mathbf{z})$ while satisfying $CSP(\mathbf{z})$.

The solutions of CSPs and COPs are computed by dedicated constraint solvers. Their efficacy and efficiency are determined by the implemented search algorithm, constraint propagation and branching. These areas are adoptable to the problem that should be solved.

[3] A similar version of this section was published in [58], reprinted with permission from "Inga Abel, Maximilian Neuner, Helmut Graeb, COPRICSI: COnstraint-PRogrammed Initial Circuit SIzing, INTEGRATION - the VLSI journal" © 2021 Elsevier.

4.4.1. Search Algorithm

CSPs are usually solved using backtracking search algorithms in form of a depth-first search (DFS) [81]. Only one solution is persecuted at a time leading to a polynomial complexity.

COPs are solved by methods based on branch-and-bound (BAB) algorithms [87]. The first solution of the problem $\mathbf{s}$ must be found by a backtracking search algorithm. The vector of target functions $\mathbf{t}(\mathbf{z})$ is added as additional constraint of form $\mathbf{t}(\mathbf{z}) > \mathbf{t}(\mathbf{s})$ to the problem, forcing the target vector of the next solution to be greater than the previous one. This step is repeated iteratively until the constraint cannot be satisfied anymore. For an effective search, constraint propagation on the objective constraint is necessary, which is described in the next section.

To increase the effectiveness of a search, a restart-based search (RBS) [88] is used. An RBS eliminates *mistakes* early in the beginning of the search tree by using a different variable/ value ordering for each restart. A mistake is a root of a subtree with no solution. A different variable/value ordering for every restart is generated through randomization. A randomization is effective if the search tree generated after a restart differs at the top significantly from the search tree before. A backtracking search algorithm is restarted after reaching a certain termination criterion, also called cutoff. The cutoff is, e.g., the number of allowed dead-ends or a fixed number of nodes the tree is allowed to contain. As more than one restart should be performed, a sequence of cutoffs $T = (t_1, t_2, t_3, ...) \subseteq \mathbb{N}$ is defined.

According to [89], this sequence can be chosen optimally if the runtime distribution of the algorithm is known. However, in most cases the runtime distribution is not known and also cannot be predicted. Then the so called luby-sequence $T_u = (1, 1, 2, 1, 1, 2, 4, 1, 1, 2, 4, 8, ...)$ is within a log factor of the optimal sequence in terms of runtime complexity. For practical usage, the sequence might not increase fast enough. However, scaling the sequence by a factor does not affect the optimality guarantee.

4.4.2. Constraint Propagation

Constraint propagation [83] is a process to make every constraint in a problem arc-consistent. A binary constraint is arc-consistent, iff every value in the domain of one variable has a corresponding value in the domain of the other variable with which the constraint is fulfilled. This definition can be extended to include non-binary constraints.

A binary constraint can be made arc-consistent by removing all values from the domain of the one variable not having a corresponding value in the domain of the other variable. By shrinking the domain of one variable, other constraints might no longer be arc-consistent. Therefore, if a domain of a variable is reduced, other constraints dependent on this variable have to be (re-)considered during constraint propagation. A classical method to obtain arc-consistency for every binary constraint of a problem is described in [90]. Extensions for non-binary constraints where made in [91].

With constraint propagation, domain sizes can be reduced and thus mistakes in the search tree are avoided. Constraint propagation is run often in the beginning of the search. By knowing which domains depend on another, constraint propagation is also run successfully during search. When a domain was reduced by branching (Sec. 4.4.3), constraint propagation should be run on its dependent constraints.

4.4.3. Branching

If constraint propagation cannot further reduce the variable domains, a new subtree has to be grown, i.e., a value has to be assigned to a variable or a variable domain has to be reduced by a new constraint. This procedure is called branching [81]. Branching consists of two tasks:

- Selecting the variable which should be considered next.
- Selecting the value or the new constraint which should be assigned to the variable.

Branching is in most cases crucial to solve a CSP/COP. A branching heuristic is optimal if the number of visited nodes is minimal. However, a method to find such an optimal heuristic has not yet been developed [81].

Variable selection methods: A good variable selection method keeps the number of visited nodes in a search tree as low as possible. Three characteristics of a variable have to be considered [81]:

- its domain size dom
- its degree deg
- its weighted degree $wdeg$

The degree deg describes the number of constraints a variable is part of. The weighted degree $wdeg$ measures the importance of a variable to lead to a valid solution. It is the degree of the variable increased by 1 for every dead-end (leave with no solution in a search tree) a constraint with the variable in can be hold responsible for. Variables with a small solution set have a higher weighted degree than variables with a large number of values leading to solutions of the problem. A good branching heuristic prefers variables with a small domain size and a small solution set (high weighted degree). The small domain size leads to a small number of nodes in the beginning of the search tree. The high weighted degree leads to fewer mistakes (subtrees without a solution) during the development of the search tree. Such a branching heuristic can be obtained by ordering the variables according to increasing $\frac{dom}{wdeg}$. It is started with the variable with the smallest $\frac{dom}{wdeg}$-value (small domain size, large weighted degree). The last variable that is considered during branching is the variable with the highest $\frac{dom}{wdeg}$-value (large domain size, small weighted degree). The variable selection method is dynamic. A new variable ordering is obtained in every branching step as new values for dom and $wdeg$ are calculated in every branching step. It has a low computational costs. Additionally, static variable selection methods exist. They have a fixed ordering generated before the start of the search. They should be problem-specific to be effective [81]. An example for a static variable selection method is described in [57].

Value assignment methods: Dynamic value assignment strategies have all a high computational effort, and therefore are not feasible [81]. In the following, only static methods of value assignments are discussed. Three major methods exist [81]: enumeration, binary choice point, and domain splitting. An enumerating value assignment method generates for each value in the domain a branch. This can lead to a search of unfeasible size. In the binary choice point method, only two branches are posted. One branch has the additional constraint that the variable must equal a certain value. To the other branch, the constraint is added that the variable is unequal to that value. Using domain splitting, the variable is not assigned a certain value, but two branches are created with an additional constraint

posted on each of them. The domain of the variable is restricted for one branch to the lower half, for the other branch to the upper half of the old domain.

4.5. Solving the Initial Sizing Problem Using Constraint Programming[4]

To improve the convergence rate of the constraint programming solver, a dynamic, problem-specific branching heuristic was developed for the initial sizing problem of analog circuits. The corresponding variable selection method (Sec. 4.5.1) and value assignment strategy (Sec. 4.5.2) are presented in the following.

4.5.1. Variable Selection Method

As stated in Sec. 4.4, a branching method is optimal when the least possible number of nodes in a search tree is visited before finding a solution. As an optimal branching method does not exist, the search tree can at least be reduced and mistakes in the search avoided by ordering the variables according to increasing $\frac{dom}{wdeg}$.

As the domain size is only calculable for finite integer values, the newly developed branching method only allows integer variables as input. The variables of the initial sizing problems are the width w_k, length l_k and drain-source current $i_{DS,k}$ of all transistors $t_k \in T$, the net voltages v_l of all nets $n_l \in N$, and all performance features $\mathbf{z}$ (Sec. 4.2.1). As manufacturing processes require finite values for w_k, l_k on a certain scale, e.g., 1μm, 0.1μm and as w_k, l_k have only linear dependencies on each other as, e.g., in (4.5), w_k, l_k can be expressed through integer values. All other variables have non-linear dependencies on other variables and therefore cannot be as easily set to be integers. Thus, auxiliary float variables are used during calculation which are converted to integers being exact to a certain magnitude. For currents this is nA, for voltages mV. For the performance features it varies depending on the unit.

As we show in the experimental results (Sec. 4.6.3), ordering the variables according to $\frac{dom}{wdeg}$ is not sufficient for the initial sizing process. Hence, this method was combined with the static branching method in [57]. The corresponding algorithm is given in Alg. 4.

Following [57], every transistor t_k is regarded as a whole having a set of variables:

$$t_k = \{w_k, l_k, i_{DS,k}, v_{S,k}, v_{G,k}, v_{D,k}\} \tag{4.53}$$

with $v_{G,k}, v_{D,k}, v_{S,k}$ being the voltage potentials of gate G, drain D and source S, which correspond to the net voltages of the connected nets. Thus, a transistor t_k can be treated as one variable composed of a set of other variables. Its domain size $dom(t_k)$ is calculated as follows:

$$dom(t_k) = \sum_{i=1}^{|t_k|} \frac{dom(d_i)}{maxdom(d_i.type)} \tag{4.54}$$

[4] A similar version of this section was published in [58], reprinted with permission from "Inga Abel, Maximilian Neuner, Helmut Graeb, COPRICSI: COnstraint-PRogrammed Initial Circuit SIzing, INTEGRATION - the VLSI journal" © 2021 Elsevier.

Algorithm 4 Dynamic branching heuristic

Require: Set of all variables $\mathcal{X}$; Set of all transistors $\mathcal{T} \subset \mathcal{X}$

```
branchBiasTransistor(T)
X_ord := sortAllVariablesAccordingToSmallestDomain(X)
for all x ∈ X_ord do
  if isTransistor(x) then
    d_x = findUnassignedVariable(x)//Assignment according to variable ordering in (4.60)
    if d_x.type == l or d_x.type == w then
      d_x.value = generateRandomNumber()
    else
      d_x.value = d_x.med()
    end if
  else
    x.value = x.med()
  end if
end for
```

with $maxdom(d_i.type)$ as the greatest domain size of the variables of the same type, as e.g., width w. This value changes dynamically during the branching process as it highly depends on constraint propagation. The weighted degree of a transistor $wdeg(t_k)$ is similarly calculated:

$$wdeg(t_k) = \sum_{i=1}^{|t_k|} wdeg(d_i) \tag{4.55}$$

The normalization of the variables' domain sizes in (4.54) is needed to prefer transistors which are highly affected by constraint propagation over transistors which are not. It lessens the effect of the different variable scalings and of large weighted degrees due to unassigned variables. An illustration is given in Table 4.2. It shows the domains (D), domain sizes (dom) and weighted degrees ($wdeg$) of two transistors t_1, t_2. The domain size dom [$clas.$] is calculated without domain normalization and corresponds to the magnitude of the domain D. For the length of transistor t_1, this is: $dom_{clas,l,t_1} = |D_{l,t_1}| = 10$. dom [$norm.$] shows the values calculated according to (4.54). Domain normalization is considered for these values. In Table 4.2, the maximum domain size of a variable type $maxdom$ is marked in bold. They are the largest domain size values for the different variable types for the two transistor. The normalized domain size of the length variable of transistor t_1 is calculated by:

$$dom_{norm,l,t_1} = \frac{dom_{clas,l,t_1}}{maxdom_l} = \frac{10}{10} = 1 \tag{4.56}$$

The normalized domain size of the length of transistor t_2 is:

$$dom_{norm,l,t_2} = \frac{dom_{clas,l,t_2}}{maxdom_l} = \frac{1}{10} = 0.1 \tag{4.57}$$

The last column of the table shows the total domain sizes of the transistors (transistor domains) and total weighted degrees of the transistors. They correspond to the sum of the

Table 4.2.: Domains and weighted degrees of two transistors during branching

Transistor		l	w	i_{DS}	v_S	v_G	v_D	Total
t_1	D	$[1,..,10]$	$[12,..,600]$	$[85820,..,121666]$	$[0,..,100]$	$[446,..,705]$	$[750,..,1000]$	-
	$dom\ [clas.]$	**10**	**589**	**35847**	**101**	**260**	**251**	37058
	$dom\ [norm.]$	1	1	1	1	1	1	6
	$wdeg$	16	26	19	15	22	8	106
t_2	D	5	$[424,..,600]$	$[85820,..,121666]$	1500	699	750	-
	$dom\ [clas.]$	1	177	**35847**	1	1	1	36028
	$dom\ [norm.]$	0.1	0.3	1	0.01	0.004	0.004	1.418
	$wdeg$	0	16	12	0	0	0	28

individual domain sizes (weighted degrees) calculated for the variables. For the normalized domain $dom\ [norm.]$, this corresponds to the solutions of equation (4.54). The total domain sizes $dom\ [clas.]$, which were calculated without domain normalization, are dominated by the large domain of the current. The current variables i_{DS,t_1}, i_{DS,t_2} of the two transistors have the largest domain sizes with a value of 35847 for both transistors. Calculating the $\frac{dom}{wdeg}$-values of the two transistors using $dom\ [clas.]$, we obtain:

$$\frac{dom(t_1)}{wdeg(t_1)} = \frac{37048}{106} \approx 446, \quad \frac{dom(t_2)}{wdeg(t_2)} = \frac{36028}{28} \approx 1287 \tag{4.58}$$

A branching heuristic ordering the variables according increasing $\frac{dom}{wdeg}$-value (Sec. 4.4.3) would prefer transistor t_1 over t_2, as t_1 has a much higher weighted degree and thus a much smaller $\frac{dom}{wdeg}$-value. However, reasoning recommends to prefer t_2 over t_1. t_2 has a width domain advantageously small having a high probability to minimize the number of visited nodes in the search tree. With the domain normalization in (4.54), this reasoning is followed during branching as with normalized domains, we obtain:

$$\frac{dom(t_1)}{wdeg(t_1)} = \frac{6}{106} \approx 0.06 > \frac{dom(t_2)}{wdeg(t_2)} = \frac{1.418}{28} \approx 0.05 \tag{4.59}$$

A branching heuristic ordering the variables according increasing $\frac{dom}{wdeg}$-value would prefer t_2 over t_1.

The proposed branching heuristic (Alg. 4) orders all variables of the initial sizing problem according to increasing $\frac{dom}{wdeg}$, taking transistor variables as one variable t_k (4.53) and calculating the transistor domain according to (4.54). For the variables of the circuit not included in the transistor variables, e.g., the performance features $\mathbf{z}$, no domain normalization is used. Hence, the domain sizes of transistors variables are significantly smaller than the domain sizes of the other variables. Thus, the transistor variables are preferred, which leads to a similar but dynamical branching heuristic as in [57].

The internal variable order of a transistor is static and identical to the ordering in [57]:

$$l_k \rightarrow w_k \rightarrow i_{DS,k} \rightarrow v_{S,k} \rightarrow v_{G,k} \rightarrow v_{D,k} \tag{4.60}$$

This ordering is chosen again according to smallest domain size and error rate. The domain size of a transistor length l_k is likely to be smaller than the domain size of w_k. After assigning l_k and w_k, $i_{DS,k}$ is set. The current domain is reduced by constraint propagation as $i_{DS,k} \sim \frac{w_k}{l_k}$ (Sec. 4.2.3). Preferring the current variable $i_{DS,k}$ over the voltage variables

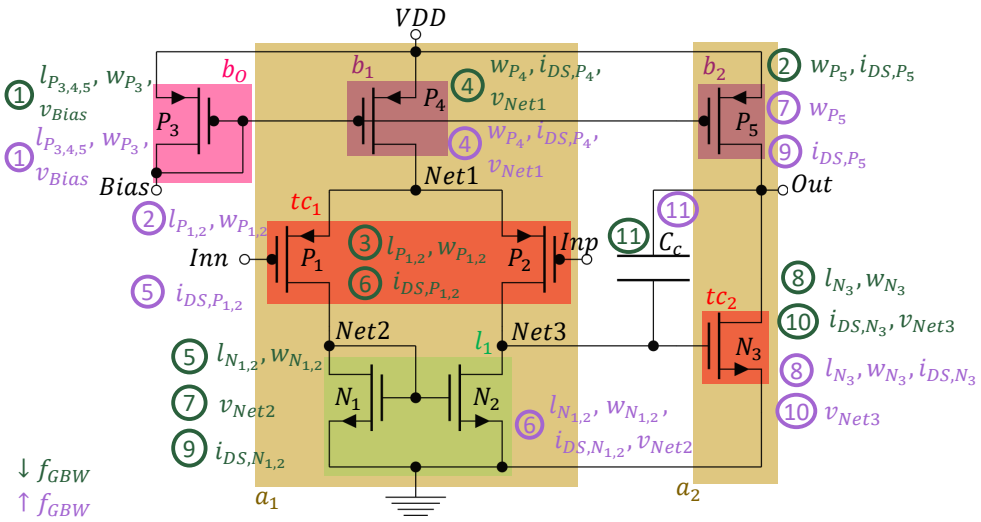


Figure 4.4.: Miller op-amp with two different branching orders

reduces the mistakes during search, as $i_{DS,k}$ is part of many more constraints than the voltage variables, e.g., KCL, performance features. The drain voltage potential $v_{D,k}$ is part of the fewest number of constraints and therefore is selected at last.

Please note that not all variables of a transistor may be assigned at once. Often, one or two variables of a transistor are assigned and then it is switched to another transistor with a smaller domain size due to processed constraint propagation. Also note that the branching algorithm (Alg. 4) assigns in the first step statically the variables of the bias input transistor, i.e., the transistor which carries the bias current (e.g., Fig. 4.4, P_3). The current of this transistor is known as it is inputted to the sizing process (Fig. 4.1) as part of the external supplies. This shrinks its domain significantly. Assigning the bias input transistor first reduces the search space of the initial sizing problem, as the domain sizes of the transistors forming the stage biases in the circuit are reduced as well. The number of solutions in the search space is not significantly affected, as the performance features are not directly dependent on the bias input transistor. Errors at the beginning of the search tree are thus avoided. Afterwards, the dynamical allocation of a variable ordering is performed. Because it is a dynamical variable ordering, the branching order can be different for different specifications.

Fig. 4.4 shows branching orders for two different specifications on a Miller op-amp. In green, a branching ordering requiring a small unity-gain bandwidth f_{GBW} is shown. In purple, a branching order for a high unity-gain bandwidth is shown. The green branching order for a smaller f_{GBW} considers the stage bias of the second stage P_5 right after the bias input transistor P_3. The branching order is different for a large unity-gain bandwidth. Here, the transistors of the transconductor of the first stage P_1, P_2 are sized after the bias input transistor. The sizes of these transistors highly affect the value of the unity-gain bandwidth.

4.5.2. Value Assignment Strategy

The values are assigned to the variables according to the binary choice point method (Sec. 4.4.3). Random numbers are assigned to the transistor widths and lengths. This

Table 4.3.: Symmetry constraints

	Telescopic op-amp	Symmetrical op-amp with high PSRR	Folded-cascode op-amp with CMFB	Complementary op-amp
HL 2: Structures	8	5	7	7
HL 3: Amplification stage subblocks	5	3	12	8
HL 4: Op-amp subblocks	-	4	-	-

allows to establish an effective restart-based search (Sec. 4.4.1). As width and length of every transistor are assigned first, an effective randomization for every restart near the beginning of the search tree is obtained. The mean domain values are assigned to all other variables. They depend directly or indirectly on transistor widths and lengths. The mean value is a good prediction of the actual value leading to less mistakes during the search.

4.6. Experimental Results

This section present results generated with the equation-based sizing tool. Sec. 4.6.1 presents the equation-based circuit models automatically synthesized for the four circuits in Fig. 4.2. Sizing results obtained with these models are given in Sec. 4.6.2. Also, sizing results for the Miller op-amp (Fig. 4.4) are shown using weak inversion as operating region for some of the transistors. Sec. 4.6.3 shows a runtime comparison of the problem-specific branching heuristic (Sec. 4.5) and a generic branching heuristic. Sec. 4.6.4 discusses the time which is needed to extend the equation-based sizing tool to support other analog circuit classes.

4.6.1. Performance Model[5]

In the following sections, the important parts of the performance models of the four circuits in Fig. 4.2 are described. All equations were generated individually and automatically using the algorithms in Sec. 4.3. Please note that the generated circuit models correspond well to the circuit models presented in analog design books [48, 50, 51, 80].

Symmetry Constraints

Table 4.3 shows the symmetry constraints derived for the four circuits in Fig. 4.2. Eight symmetry constraints for basic structures were derived for the telescopic op-amp (Fig. 4.2a). This is identical to the number of current biases in the circuit. The large number of symmetry constraints for the amplification stage subblock level in the folded-cascode op-amp with Common-Mode FeedBack (CMFB) results from the CMFB stage in which both differential pairs must be identical. As the two second stages $a_{2,1}, a_{2,2}$ in the symmetrical op-amp with high PSRR (Fig. 4.2c) must be symmetric, four symmetry constraint were derived for HL 4 for this circuit.

[5]A similar version of this section was published in [59], reprinted with permission from "Inga Abel, Maximilian Neuner, Helmut Graeb, A Hierarchical Performance Equation Library for Basic Op-Amp Design, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems" © 2021 IEEE.

Functional Block Constraints

As a cascode current mirror forms one load part of the telescopic op-amp (Fig. 4.2a), its widths are restricted by the corresponding behavior constraint (4.13). Furthermore, as the telescopic op-amp has a second stage, the output voltages of the first stage, i.e., the voltage potentials of the nets n_5,n_8, must be equal (4.16).

To make the combination of folded-cascode op-amp and CMFB circuit work, a functional block constraint on the fifth hierarchy level not mentioned before must be generated for the folded-cascode op-amp with CMFB (Fig. 4.2b). The unity-gain bandwidth of the CMFB circuit must be bigger than the one of the op-amp, such that the CMFB circuit is faster.

$$f_{GBW,CMFB} > f_{GBW,op-amp} \tag{4.61}$$

The unity-gain bandwidth is calculated according to (4.42) treating the CMFB stage as a first stage.

As the complementary op-amp (Fig. 4.2d) has a complementary first stage, the functional block constraints for HL 3 are generated restricting the first stage transconductor to have equal transconductances, and the stages biases to produce equal currents.

Performance Equations

In the following paragraphs, the performance equations of the four circuits in Fig. 4.2 are presented. We focus on the differences between the four circuits.

Quiescent power: For the telescopic op-amp (Fig. 4.2a), the currents flowing into the positive supply voltage rail are considered to calculate the power consumption, while for the other three circuits also the bias currents of the circuit must be considered as it is applied to NMOS transistors. The equation for the quiescent power of the telescopic op-amp is:

$$z_{QP} = (v_{VDD} - v_{VSS}) \cdot (|i_{DS,P_7}| + |i_{DS,P_9}| + |i_{DS,P_5}| + |i_{DS,P_8}|) \tag{4.62}$$

and for the quiescent power of the symmetrical op-amp with high PSRR (Fig. 4.2c):

$$z_{QP} = (v_{VDD} - v_{VSS}) \cdot (|i_{DS,P_3}| + |i_{DS,P_1}| + |i_{DS,P_2}| + |i_{DS,P_4}| + |i_{DS,P_7}| + |i_{DS,N_7}|) \tag{4.63}$$

Common-mode input voltage: For the telescopic op-amp (Fig. 4.2a), the maximum input voltage is set by the path over the first stage stage bias:

$$z_{v_{cm,max}} = v_{cm,b_{s,1}} = v_{VDD} + v_{GS,P_1} + v_{GS,P_5} - v_{th,p} \tag{4.64}$$

Please note that the gate-source voltages have negative values.

For the symmetrical op-amp with high PSRR (Fig. 4.2c) and the folded-cascode op-amp with CMFB (Fig. 4.2b), $z_{v_{cm,max}}$ is set by the path over the load. For the folded-cascode op-amp with CMFB, this is:

$$z_{v_{cm,max}} = v_{cm,l_1} = v_{VDD} + v_{th,n} + v_{GS,P_3} - v_{th,p} \tag{4.65}$$

In the telescopic op-amp (Fig. 4.2a), the load defines the minimum input voltage. As higher minimum saturation voltages must be respected, the load path with the two diode transistors N_1, N_2 is selected:

$$z_{v_{cm,min}} = v_{ss} + v_{th,p} - (v_{GS,P_3} - v_{th,n}) + v_{GS,N_1} + v_{GS,N_2} \tag{4.66}$$

In the other circuits, the minimum input voltage is restricted by the stage bias of the first stage, which leads to similar equations as in (4.64) with the negative supply voltage as input. No equations are generated for the complementary op-amp (Fig. 4.2d), as it is assumed to allow all values as input voltage.

Output voltage: In the telescopic op-amp (Fig. 4.2a) and the symmetrical op-amp with high PSRR (Fig. 4.2c), the output voltage is restricted by one transistor on each path of the output stage. The output voltage equations for the telescopic op-amp are:

$$\begin{aligned} z_{v_{out,max}} &= v_{VDD} + v_{GS,P_6} - v_{th,p} \\ z_{v_{out,min}} &= v_{VSS} + v_{GS,N_5} - v_{th,n} \end{aligned} \tag{4.67}$$

For the folded-cascode op-amp with CMFB (Fig. 4.2b) and the complementary op-amp (Fig. 4.2d), the output voltage is restricted by the load parts of the first stage, as the first stage is also the output stage. For each path, two transistors must be considered. For the folded-cascode op-amp with CMFB, the output voltage is restricted by:

$$\begin{aligned} z_{v_{out,max}} &= v_{VDD} + v_{GS,P_4} - v_{th,p} + v_{GS,P_2} - v_{th,p} \\ z_{v_{out,min}} &= v_{VSS} + v_{GS,N_8} - v_{th,n} + v_{GS,N_6} - v_{th,n} \end{aligned} \tag{4.68}$$

Common-mode rejection ratio: The common-mode rejection ratio is calculated for the telescopic op-amp (Fig. 4.2a) and the symmetrical op-amp with high PSRR (Fig. 4.2c). To calculate CMRR of the telescopic op-amp, the open-loop gain of the first stage is needed. Using (4.25), (4.21), (4.19), we obtain:

$$A_{D0,1} = \frac{gm_{P_1}}{\frac{gd_{P_1} \cdot gd_{P_3}}{gm_{P_3}} + \frac{gd_{N_1} \cdot gd_{N_3}}{gm_{N_1}}} \tag{4.69}$$

N_1 is the load transistor chosen for the CMRR calculation as its gate is connected to an output of the first stage, which leads to following CMRR equation for the telescopic op-amp:

$$z_{CMRR} = 2A_{D0,1} \cdot \frac{gm_{N_1}}{gd_{P_5}} \tag{4.70}$$

The first and second stage gain must be considered to calculate the CMRR of symmetrical op-amps (4.40). For the symmetrical op-amp with high PSRR (Fig. 4.2c), these are:

$$A_{D0,1} = \frac{gm_{N_1}}{gd_{N_1} + gm_{P_1}}, \quad A_{D0,2} = \frac{gm_{P_4}}{\frac{gd_{P_6} + gd_{P_4}}{gm_{P_6}} + gd_{N_5}} \tag{4.71}$$

The equation of the CMRR is:

$$z_{CMRR} = 2A_{D0,1} \cdot A_{D0,2} \frac{gm_{P_2}}{gd_{N_3}} \tag{4.72}$$

Unity-gain bandwidth: The unity-gain bandwidth is calculated similarly for the telescopic op-amp (Fig. 4.2a), for the folded-cascode op-amp with CMFB (Fig. 4.2b) and for the complementary op-amp (Fig. 4.2d). For the telescopic op-amp, it is:

$$z_{f_{GBW}} = \frac{gm_{P_1}}{2\pi C_{n_8}} \tag{4.73}$$

In the complementary op-amp (Fig. 4.2d), both NMOS and PMOS differential pairs must be considered to calculate the transconductance of the first stage transconductor (4.20).

In the symmetrical op-amp with high PSRR (Fig. 4.2c), the second stage must be considered to calculate the unity-gain bandwidth (4.43). The equation becomes:

$$z_{f_{GBW}} = \frac{A_{D0,1} \cdot gm_{P_4}}{2\pi C_{n_5}} \tag{4.74}$$

Open-loop gain: The open-loop gain is calculated by the multiplication of the gains of the stages. Two stages must be considered in the telescopic op-amp (Fig. 4.2a), three stages in the symmetrical op-amp with high PSRR (Fig. 4.2c). As the folded-cascode op-amp with CMFB (Fig. 4.2b) consists of one stage only, its open-loop gain is the gain of the first stage.

In the complementary op-amp (Fig. 4.2d), two gate-connected couples exist. The open-loop gain is therefore calculated by:

$$z_{A_{D0}} = \frac{gm_{N_4} + gm_{P_4}}{\frac{gd_{P_8} \cdot (gd_{P_6} + gd_{N_4})}{gm_{P_8}} + \frac{gd_{N_8} \cdot (gd_{N_6} + gd_{P_4})}{gm_{N_8}}} \tag{4.75}$$

Slew rate: In the telescopic op-amp (Fig. 4.2a), the first stage and the second stage bias current must be considered for the slew rate:

$$z_{SR} = min\{\frac{|i_{DS,P_5}|}{C_{n_8}}, \frac{|i_{DS,P_6}|}{C_{n_{out}}}\} \tag{4.76}$$

Please note that the capacitance of net n_8 is mainly influenced by the compensation capacitor c_C, the capacitance of net n_{out} by the load capacitor c_L.

In the symmetrical op-amp, the second stage and the third stage are considered for slew rate calculation. Twice the current of the second stage is considered.

$$z_{SR} = min\{\frac{2 \cdot |i_{DS,N_5}|}{C_{n_5}}, \frac{|i_{DS,N_6}|}{C_{n_{out}}}\} \tag{4.77}$$

In the folded-cascode op-amp with CMFB (Fig. 4.2b), the bias currents of the first stage differential pair as well as the gate connected couple must be considered. This leads to:

$$z_{SR} = \frac{min\{|i_{DS,N_4}|, |i_{DS,P_4}|\}}{C_{n_{out}}} \tag{4.78}$$

The same considerations must be made for the complementary op-amp (Fig. 4.2d). In addition, the PMOS and NMOS stage biases are of interest:

$$z_{SR} = \frac{min\{(|i_{DS,N_2}| + |i_{DS,P_2}|), (|i_{DS,N_6}| + |i_{DS,P_6}|)\}}{C_{n_{out}}} \tag{4.79}$$

Phase margin: Two non-dominant poles are identified for the telescopic op-amp (Fig. 4.2a): one pole for the first stage and one for the second stage. The compensation capacitor brings a positive zero along. Hence, the automatically generated equation for the phase margin is:

$$z_{PM} = \frac{\pi}{2} - \text{atan}(\frac{f_{GBW}}{f_{ndp,a_1}}) - \text{atan}(\frac{f_{GBW}}{f_{ndp,a_2}}) - \text{atan}(\frac{f_{GBW}}{f_{pz}}) \tag{4.80}$$

In the symmetrical op-amp with high PSRR (Fig. 4.2c), three non-dominant poles arise: the first stage non-dominant pole, the non-dominant pole evoked by the compensation capacitor in the third stage and the non-dominant pole of the cascode transconductors in the second stages. The compensation capacitor also leads to a positive zero. The equation for the phase margin is:

$$z_{PM} = \frac{\pi}{2} - \text{atan}(\frac{f_{GBW}}{f_{ndp,a_1}}) - \text{atan}(\frac{f_{GBW}}{f_{ndp,a_3}}) - \text{atan}(\frac{f_{GBW}}{f_{ndp,a_{c,2}}}) - \text{atan}(\frac{f_{GBW}}{f_{pz}}) \tag{4.81}$$

For the folded-cascode op-amp with CMFB (Fig. 4.2b), the phase margins of the first stage and the CMFB circuit must be calculated. As the phase margin of the CMFB circuit is restricted by the non-dominant poles of the first stage and the CMFB stage, this phase margin is the most restrictive one.

In the complementary op-amp (Fig. 4.2d), two non-dominant poles of the first stage must be calculated, respecting the two differential pairs.

4.6.2. Sizing Results

The instantiated equations and constraints are automatically given to the embedded constraint programming solver GeCode [92]. A restart-based branch-and-bound algorithm is used as search algorithm. As cutoff sequence, a luby-sequence with a scaling factor of 10 is used. The tool needs a few seconds to find the first initial sizing for a circuit. The sizing is optimized towards higher performance safety margins for a selected set of performance features $\mathbf{z}_{opt}$,

$$\mathbf{z}_{opt}^T = \{z_{A_{D_0}}, z_{SR}, z_{f_{GBW}}, z_{QP}, z_D\} \tag{4.82}$$

with $z_{A_{D_0}}$ the open-loop gain, z_{SR} the slew rate, $z_{f_{GBW}}$ the unity-gain frequency, z_{QP} the power consumption and z_D the transistor gate-area. The target vector is defined as scalar:

$$t(\mathbf{y}) = \sum_{i=1}^{|\mathbf{z}_{opt}|} \tilde{z}_i(\mathbf{y}) \tag{4.83}$$

were $\tilde{z}_j(\mathbf{y})$ is a normalized performance feature according to:

$$\tilde{z}_j(\mathbf{y}) = \frac{z_j(\mathbf{y}) - min(z_j)}{dom(z_j)} \tag{4.84}$$

for a performance feature to be maximized and

$$\tilde{z}_j(\mathbf{y}) = \frac{max(z_j) - z_j(\mathbf{y})}{dom(z_j)} \tag{4.85}$$

for a performance feature to be minimized. $\mathbf{y}$ are the variables of the sizing problem generated with the HPEL (Sec. 4.2.1). The optimization is run for one more minute. After one minute, the improvement slowed down significantly in the experiments, therefore the optimization loop has been set to run for one minute overall.

The domains of the transistor dimensions are $D_L = [1, ..., 10]$ μm, $D_W = [1, ..., 600]$ μm with a discretization of 1μm. The domains of the net voltage potentials are $[VSS, .., VDD]$ with a discretization of 1mV and the domains of the transistor drain-source currents are [-10,..,10] mA with a discretization of 1nA.

Table 4.4 presents sizing values for the circuits in Fig. 4.2 and Fig. 4.4, using the specifications in Table 4.5. The performance values calculated with the performance models and results from circuit simulation are included in these tables. All sizing results were generated with a 0.25μm PDK. For the Miller op-amp (Fig. 4.4), the supply voltage and current are low 2V and 1μA respectively, such that the transistors of the transconductors, namely P_1, P_2, N_3, and load, N_1, N_2, are set to operate in weak inversion. For the other circuits, the supply voltage is 5V, the bias current 10μA. The load capacity of the circuit is set to be 20pF. The last column of Table 4.5 shows the average deviations between the performance values calculated with the sizing tool and simulation results of 20 different circuits (Table 4.6). The specifications were adjusted for each circuit, such that the circuit could fulfill the specifications.

The deviations between the calculated and simulated values is 19% on average. This meets the requirement of analog designers who expect a 20% - 30% deviation between the Shichman-Hodges model and full circuit simulation. The highest deviation occurs for the unity-gain bandwidth with a deviation of over 30%. The unity-gain bandwidth is often overestimated. For the telescopic op-amp (Fig. 4.2a) and the symmetrical op-amp with high PSRR (Fig. 4.2c), for example, it does not fulfill the specifications in simulations. A reason for the high deviation is the strong dependency of the unity gain-bandwidth on the transconductance of the input transistors of the first stage. The unity-gain bandwidth depends linearly on it (4.42). To maximize the gain, the transconductance of the input transistors is often overestimated using the Shichman-Hodge model leading to a smaller gain and unity-gain bandwidth in simulations.

Table 4.4.: Dimensions for the circuits in Fig. 4.2 and Fig. 4.4

Variable	Value
$w_{P_1} = w_{P_2}$	172μm
$w_{P_3} = w_{P_4}$	27μm
w_{P_5}	247μm
w_{P_6}	515μm
w_{P_7}	7μm
w_{P_8}	7μm
w_{P_9}	43μm
$w_{N_1} = w_{N_2}$	90μm
$w_{N_3} = w_{N_4}$	90μm
w_{N_5}	130μm
w_{N_6}	269μm
w_{N_7}	166
$l_{P_1} = l_{P_2}$	9μm
$l_{P_3} = l_{P_4} = l_{P_8}$	4μm
$l_{P_5} = l_{P_6} = l_{P_7} = l_{P_9}$	3μm
$l_{N_1} = l_{N_2}$	1μm
$l_{N_3} = l_{N_4}$	1μm
l_{N_5}	1μm
$l_{N_6} = l_{N_7}$	9μm
c_c	6.4pF

(a) Telescopic op-amp

Variable	Value
$w_{N_1} = w_{N_2}$	8μm
w_{N_3}	56μm
$w_{N_4} = w_{N_5}$	205μm
w_{N_6}	460μm
w_{N_7}	23μm
$w_{P_1} = w_{P_2}$	5μm
$w_{P_3} = w_{P_4}$	15μm
$w_{P_5} = w_{P_6}$	35μm
w_{P_7}	287μm
$l_{N_1} = l_{N_2}$	3μm
$l_{N_3} = l_{N_7}$	6μm
$l_{N_4} = l_{N_5}$	9μm
l_{N_6}	1μm
$l_{P_1} = l_{P_2} = l_{P_3} = l_{P_4} = l_{P_7}$	2μm
$l_{P_5} = l_{P_6}$	2μm
c_c	4.5pF

(b) Symmetrical op-amp with high PSRR

Variable	Value
$w_{N_1} = w_{N_2}$	548μm
w_{N_3}	5μm
w_{N_4}	290μm
$w_{N_5} = W_{N_6}$	218μm
$w_{N_7} = w_{N_8}$	30μm
$w_{N_9} = w_{N_{10}}$	141μm
$w_{N_{11}} = w_{N_{12}} = w_{N_{13}} = w_{N_{14}}$	84μm
$w_{P_1} = w_{P_2}$	175μm
$w_{P_3} = w_{P_4}$	143μm
$w_{P_5} = w_{P_6}$	55μm
$l_{N_1} = l_{N_2}$	8μm
$l_{N_3} = l_{N_4} = l_{N_5} = l_{N_6} = l_{N_9} = l_{N_{10}}$	3μm
$l_{N_7} = l_{N_8}$	2μm
$l_{N_{11}} = l_{N_{12}} = l_{N_{13}} = l_{N_{14}}$	1μm
$l_{P_1} = l_{P_2}$	2μm
$l_{P_3} = l_{P_4} = l_{P_5} = l_{P_6}$	1μm

(c) Folded-cascode op-amp with CMFB

Variable	Value
w_{N_1}	13μm
w_{N_2}	306μm
$w_{N_3} = w_{N_4}$	79μm
$w_{N_5} = w_{N_6}$	39μm
$w_{N_7} = w_{N_8}$	32μm
w_{N_9}	13μm
w_{P_1}	7μm
w_{P_2}	168μm
$w_{P_3} = w_{P_4}$	378μm
$w_{P_5} = w_{P_6}$	66μm
$w_{P_7} = w_{P_8}$	104μm
$l_{N_1} = l_{N_2} = l_{N_9}$	4μm
$l_{N_3} = l_{N_4}$	1μm
$l_{N_5} = l_{N_6}$	5μm
$l_{N_7} = l_{N_8}$	5μm
$l_{P_1} = l_{P_2}$	3μm
$l_{P_3} = l_{P_4}$	1μm
$l_{P_5} = l_{P_6}$	3μm
$l_{P_7} = l_{P_8}$	3μm

(d) Complementary op-amp

Variable	Value
$w_{P_1} = w_{P_2}$	337μm
w_{P_3}	5μm
w_{P_4}	8μm
w_{P_5}	98μm
$w_{N_1} = w_{N_2}$	27μm
w_{N_3}	585μm
$l_{P_1} = l_{P_2}$	1μm
$l_{P_3} = l_{P_4} = l_{P_5}$	3μm
$l_{N_1} = l_{N_2}$	1μm
l_{N_3}	1μm
c_c	4.5pF

(e) Miller op-amp

Table 4.5.: Performance values of the (a) telescopic op-amp (b) symmetrical op-amp with high PSRR (c) folded-cascode op-amp with CMFB (d) complementary op-amp (e) Miller op-amp

Constraints	Specifications					Sizing tool					BSIM3v3					Average deviation
	(a)	(b)	(c)	(d)	(e)	(a)	(b)	(c)	(d)	(e)	(a)	(b)	(c)	(d)	(e)	
Bias current (μA)	10	10	10	10	1	10	10	10	10	1	10	10	10	10	1	-
Supply volt. (V)	5	5	5	5	2	5	5	5	5	2	5	5	5	5	2	-
Gate-area (10^3 μm^2)	≤ 15	≤ 10	≤ 15	≤ 5	≤ 3	5.8	5.5	13.4	4.5	1.6	-	-	-	-	-	-
Quiescent power (mW)	≤ 10	≤ 15	≤ 15	≤ 5	≤ 0.1	5.8	4	10	5	0.044	6.1	4.5	11	4.4	0.047	13%
Max. common-mode input voltage (V)	≥ 3	≥ 3	≥ 3	-	≥ 1.2	3.3	4.3	4.5	-	1.4	4.4	4.3	4.4	-	1.3	-
Min. common-mode input voltage (V)	≤ 2	≤ 2	≤ 2	-	≤ 0.2	0	0.8	0.9	-	0.1	0.1	0.7	1	-	0.1	-
Max. output voltage (V)	≥ 4	≥ 4	≥ 3.5	≥ 3.5	≥ 1.8	4.5	4.5	4	3.5	1.8	4.5	4.4	4.1	3.8	1.8	-
Min. output voltage (V)	≤ 1	≤ 1	≤ 1	≤ 1.5	≤ 0.5	0.3	0.1	0.9	1.4	0.1	0.2	0.2	1	0.5	0.2	-
CMRR (dB)	≥ 90	≥ 90	≥ 80	≥ 70	≥ 80	130	95	122	133	108	146	142	118	136	114	13%
Unity-gain bandwidth (MHz)	≥ 7	≥ 7	≥ 10	≥ 10	≥ 0.5	10	10.3	10	28	1	6.5	6.8	10.5	19	0.7	36%
Open-loop gain (dB)	≥ 80	≥ 80	≥ 70	≥ 80	≥ 80	120	100	75	84	98	93	97	71	86	88	15%
Slew rate ($\frac{V}{\mu s}$)	≥ 15	≥ 10	≥ 15	≥ 15	≥ 0.2	28	15	24.5	23	0.35	22	11	19	20	0.32	14%
Phase margin (°)	≥ 60	≥ 60	≥ 60	≥ 60	≥ 60	60	61	82	62	63	67	59	83	57	67	12%
Average deviation of all perf. values	-	-	-	-	-	-	-	-	-	-	23%	23%	9%	15%	14%	19%

Other specifications which are not fulfilled after simulations are the phase margin requirements of the symmetrical op-amp with high PSRR (Fig. 4.2c) and the complementary op-amp (Fig. 4.2d). However, the deviation between the simulation and calculated value is very small, 3% respectively 9%. The equation-based model of the phase margin is quite accurate. All other specifications are fulfilled by the calculated and simulated performance values for the four circuits. Please note that, if a topology cannot fulfill given specifications, the initial sizing tool stops after one minute of search without returning a sized netlist. Therefore, the initial sizing tool is also a good indicator to check if a chosen topology is able to fulfill a given set of specifications.

Chapter 5 shows additional sizing results obtained with the HPEL. It presents the synthesis tool featuring thousands of different op-amp topologies using the HPEL to evaluate the topologies. Sizing results for 100 different topologies are compared. The average deviation is again between 20% - 30% and thus meets the expectation of designers. Please note that equation-based sizing mainly aims at initial sizing of a circuit. As in the manual design process, basic transistor models are used to generate the transistor sizes such that further manual or numerical optimization of the circuit follows.

Fig. 4.5 compares the Shichman-Hodges model used in the HPEL and the BSIM3v3 model used in simulation. It shows the transconductance gm, the output conductance gd and the drain-source current i_{DS} of a transistor for different v_{GS}-values obtained with the two models. The transistor width and length are set to 10μm, 1μm respectively. The drain-source voltage was set to be 1.5V, such that the transistor operates in saturation with strong inversion, a common working region in analog circuits. For small v_{GS} values, the two transistor models correspond well. Higher v_{GS}-values lead to deviations. Keeping v_{GS} small hence leads to accurate performance results using the HPEL. Future work is on integrating more complex

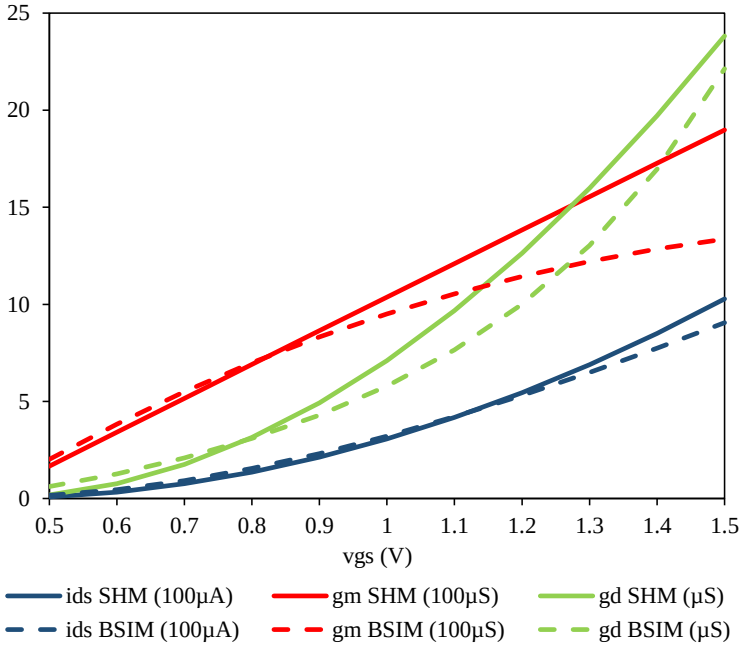


Figure 4.5.: Transistor parameter for different v_{GS}-values

transistor models, such as the EKV model, into the HPEL. The EKV model has a low complexity compared to BSIM3v3, but features a good accuracy in all transistor regions.

4.6.3. Runtime Comparison of Branching Heuristics[6]

Table 4.6 shows the CPU times required to find the first solution of the initial sizing problem for 20 different circuits using two different branching heuristics. The standard branching heuristic sorts all variables v_k of the sizing problem according to $\frac{dom(v_k)}{wdeg(v_k)}$ without considering transistor dependencies. The value assignment strategy is similar to the one explained in Sec. 4.5.2 using a random number assignment to widths and lengths, and a mean domain value assignment for all other variables. The problem-specific branching heuristic corresponds to the branching heuristic described in Sec. 4.5.

The circuits are clustered according to their characteristic first stage. The circuits 1 - 5 have a simple first stage similar to the first stage shown in Fig. 4.4. One- and two-stage variants were tested including two-stage op-amps with different load types. Regardless of the branching heuristic, the different load structures did not influence the convergence time significantly. The convergence time of both branching heuristics increased slightly for the two-stage op-amp when forcing a subset of transistors to operate in weak inversion. However, it is still in the area of seconds.

The circuits 6 - 10 have a folded-cascode first stage similar to the first stage in Fig. 4.2b. However, they all have a single output. Regardless, if the folded-cascode op-amp had a second stage and thus 20 transistors, if the resolution of the widths and lengths of the transistors was set to be higher, or if further transistors were added to the topology to enhance the slew rate, the convergence time was around 1s for both branching heuristics. Forcing transistors

[6] A similar version of this section was published in [58], reprinted with permission from "Inga Abel, Maximilian Neuner, Helmut Graeb, COPRICSI: COnstraint-PRogrammed Initial Circuit SIzing, INTEGRATION - the VLSI journal" © 2021 Elsevier.

Table 4.6.: Runtime comparison of two different branching heuristics: CPU times needed to find one solution

#	Circuit	Standard	Problem-specific
1	Simple op-amp [48]	<1s	<1s
2	Two-stage op-amp (Fig. 4.4) [48]	<1s	<2s
3	Two-stage op-amp with one load transistor [51]	<1s	<1s
4	Low-power two-stage op-amp [51]	<5s	<2s
5	Two-stage op-amp with transistors in weak inversion [48]	<15s	<20s
6	Folded-cascode op-amp [48]	<1s	<1s
7	Two-stage folded-cascode op-amp [93]	<1s	<1s
8	Folded-cascode op-amp with 0.1 mum resolution [48]	<5s	<1s
9	Folded-cascode op-amp with transistors in weak inversion [48]	<1s	<1s
10	Folded-cascode op-amp with slew-rate enhancer [50]	<5s	<5s
11	Telescopic op-amp without bias [49]	<1s	<1s
12	Telescopic op-amp with standard bias [49]	<5s	<5s
13	Telescopic op-amp with inner bias (Fig. 4.2a) [49]	<1s	<5s
14	Symmetrical op-amp [48]	<1s	<1s
15	Cascode-symmetrical op-amp [48]	<5s	<1s
16	Symmetrical op-amp with high PSRR (Fig. 4.2c) [48]	<10min	<5s
17	Symmetrical op-amp with cross-coupled gain enhancement [94]	<60s	<6s
18	Complementary amplifier (Fig. 4.2d) [95]	>1day	<10s
19	Fully-diff. op-amp with CMFB (Fig. 4.2b) [96]	>1day	<25s
20	Fully-diff. two-stage op-amp with RC-CMFB [97]	<1s	<5s
	Average duration for all circuit	-	<5s

to operate in weak inversion did not influence the convergence behavior. The convergence time of circuit 8 shows that a finer discretization has a minor influence on the runtime. Hence, the transistor sizes can be scaled down for processes with smaller channel length.

The circuits 11 - 13 are telescopic op-amps similar to the circuit in Fig. 4.2a. They differ in their bias circuit. Circuit 11 has only one diode-connected transistor as bias which supplies the first and second stage bias with the required voltages. In circuit 12 and circuit 13, the bias consists of at least five transistors, such that no outer voltage supply is required. The bias circuits differ in the supply of the first stage cascode pair. In circuit 12, the cascode pair is supplied by a diode-connected transistor with its source connected to ground. In circuit 13, the cascode pair of the first stage is supplied by a diode-connected transistor with its source pin connected to the sources of the differential pair. The results show that the topology of the bias does not influence the convergence time.

The circuits 14 - 17 are different symmetrical op-amp topologies similar to the circuit in Fig. 4.2c. For the circuits 16 and 17, the standard branching heuristic is significantly slower than the problem-specific branching heuristic. Both circuits have a high structural complexity. Circuit 16 (Fig. 4.2c) has a third stage a_3 while the other three symmetrical op-amps have two stages. In circuit 17, a cross-coupled pair is added to the normal symmetrical op-amp load. The behavioral changes coming along with these added structural blocks slow down the standard branching heuristic significantly.

On the complementary op-amp (circuit 18, Fig. 4.2d) and fully-differential op-amp with CMFB (circuit 19, Fig. 4.2b,), we can see the advantages of the problem-specific branching heuristic as it still finds a solution in less than one minute for both circuits, whereas the standard branching heuristic was unsuccessful due to the high structural complexity of the circuits. For the fully-differential two-stage op-amp (circuit 20), both branching heuristics find a solution in few seconds.

The average convergence time for all circuits was less than 5s using the problem-specific branching heuristic.

Table 4.7.: Runtime of the dynamical branching heuristic for three different specifications

Circuit	Spec. 1	Spec. 2	Spec. 3
Two-stage op-amp (Fig. 4.4)	1.3s	5s	1.2s
Folded-cascode op-amp with CMFB (Fig. 4.2b)	1.3s	53s	11s

Table 4.7 shows the runtime of the dynamical branching heuristic (Sec. 4.5) for three different specifications applied to the Miller op-amp (Fig. 4.4) and the folded-cascode op-amp with CMFB (Fig. 4.2b). Spec. 1 are common specifications which should be fulfilled easily by the chosen topologies. Spec. 2 are less common requiring a high unity-gain bandwidth and a small area. Spec. 3 are even more strict requiring, in addition to the high unity-gain bandwidth and small area, a small quiescent power and a high slew rate. For Spec. 2, the runtime is the highest, for Spec. 1 the lowest. For Spec. 3, the runtime is in-between the two other runtimes. While many solutions exist for Spec. 1 and thus, the algorithm quickly finds one, there are less solutions for Spec. 2 such that more time is needed to find one. Spec. 3 requires a trade-off between small power consumption and high slew rate, which reduces the search space of the algorithms early during search and hence speeds up the solution finding process compared to Spec. 2.

4.6.4. General Runtime Behavior and Development Time

The runtime of functional block analysis (Chapter 3) and the algorithms to set up the equation-based description of a circuit is in the area of ms. As the branching heuristic is dynamic and thus depends on intermediate search results, we can only give the runtime of the whole search algorithm of the constraint programming solver. This runtime depends on the complexity of the topologies and varies between one second and one minute searching for the first solution (Table 4.6). For optimization using a Branch-And-Bound (BAB) algorithm, we have chosen a search duration of one minute. For most circuits, no significantly better sizings with regard to the performance features were found after this time period. Thus, the runtime of the whole sizing method is dominated by the runtime of the search algorithm and is approximately one minute.

The concept of the sizing method can be extended to include additional op-amp topologies and other analog circuit classes. This requires an extension of the functional block decomposition method in Chapter 3 as well as an extension of the HPEL (Sec. 4.2) and the corresponding algorithms. A cross-coupled pair for example is frequently part of an oscillator or comparator circuit. Its formalized structural description would be added to the functional block decomposition method in Chapter 3. Its behavior equations would be added to the HPEL (Sec. 4.2). As the problem-specific branching heuristic (Sec. 4.5) is dynamic, no further development is needed when the functional block decomposition method or the HPEL is extended. The time needed to add a new functional block to the functional block decomposition method and the HPEL highly depends on its transistor structures and its contained functional subblocks. Because of the hierarchical structuring, many structural description in the functional block decomposition method are reusable. The same applies for the equation models in the HPEL. Thus, the inclusion of new blocks is easy and fast. It varies between half a working day and a few days.

5. Structural Synthesis by Functional Block Composition

This chapter presents a structural synthesis method based on the functional block decomposition method in Chapter 3 and the sizing method in Chapter 4. Based on a hierarchical functional block composition graph, op-amp topologies are automatically created for given specifications. They are sized and evaluated using the sizing method in Chapter 4, which was enhanced for op-amp synthesis. The method supports single-output, fully-differential and complementary op-amps. Thousands of different op-amp topologies can be sized and evaluated.

The following sections show how the functional blocks in Chapter 3 can be synthesized based on their description with generic algorithms (Secs. 5.1, 5.2 and 5.3). Sec. 5.4 discusses the resulting composition graph. The corresponding synthesis algorithm is explained (Sec. 5.4.2). The enhancements on the sizing method are given (Sec. 5.4.3). Sec. 5.5 discusses how the runtime of the algorithm can be reduced using multi-threading strategies. Experimental results for seven different specification sets are given in Sec. 5.6.

5.1. Synthesis of Functional Blocks Except Op-Amp Bias[1]

The hierarchical structure of functional blocks in Chapter 3 allows the synthesis of structural implementations of a functional bock based on its subblocks. This section presents a new generic algorithm which covers all blocks in Chapter 3 except the op-amp bias b_O. A separate algorithm to synthesize the op-amp bias is presented in Sec. 5.2.

5.1.1. Data Structure

A generic approach to the synthesis of op-amp functional blocks requires the transition from a functional, i.e., behavioral, description of a block, to a transistor implementation, i.e., to a structural description, of a block. In this transition, different and new pins may arise. In this work, this transition is implemented by representing functional blocks as instances with specific sets of pins. The set of pins varies for different functional block (implementation) types. Each functional block type has its own specific set of pins. The functional block *voltage bias* for instance has two implementation types, i.e., simple and cascode (Fig. 5.1). If the voltage bias is simple, its instance has three pins: $in, out_1, source$. If the voltage bias is cascode, its instance has two additional pins: $inner, out_2$. The pins define the generic pin sets for the two implementation types. They cover all possible implementation sets, even if an implementation as for instance *vr1* happens to connect two pins. This data structure provides exchangeability and flexibility in the synthesis process.

[1] A similar version of this section was published in [62], reprinted with permission from "Inga Abel, Helmut Graeb, FUBOCO: Structure Synthesis of Basic Op-Amps by FUnctional BlOck COmposition, ACM Trans. Des. Autom. Electron. Syst." © 2022 ACM.

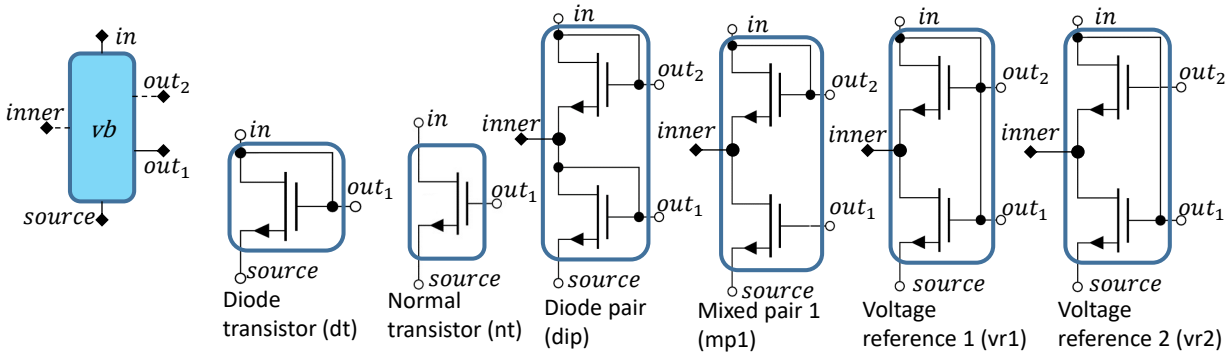


Figure 5.1.: Voltage bias instance and corresponding structural implementations

5.1.2. Generic Algorithm to Synthesize a Functional Block (Except Op-Amp Bias)

Alg. 5 creates for every functional block FB_{new} in Chapter 3 (but the op-amp bias) a set of structural implementations (instances) S_{new}.

Input

The input of the algorithm is defined in Fig. 5.2 and comprises the following sets.

Structural implementations $S_1, S_2, .., S_i$ of the functional subblocks of FB_{new}: i is the number of functional subblocks that are combined to build FB_{new}. E.g., a cascode voltage bias in Fig. 5.1 consists of two functional blocks, hence there are two sets S_1, S_2. Each set consists of normal and diode transistors $nt \in NT, dt \in DT$ having the same doping Φ ($S_1 : NT_\Phi, DT_\Phi; S_2 : NT_\Phi, DT_\Phi;$). Further input defines which combinations of these are allowed in a cascode voltage bias.

Characteristic connections R_c state how the instances in $S_1, .., S_i$ are connected to each other. This input is optional as no characteristic connections are provided for functional blocks consisting of one instance, e.g., simple voltage bias. To create a cascode voltage bias, R_c contains that the drain of a transistor $s_1 \in S_1$ must be connected to the source of a transistor $s_2 \in S_2$ ($R_c : s_1.drain \leftrightarrow s_2.source$).

Functional block rules $\mathcal{R}_f$ define wanted and unwanted connections independent of the characteristic connections. This is used to verify that a combination of different instances of functional subblocks is an implementation of FB_{new}. E.g., $\mathcal{R}_f$ of a cascode voltage bias (Fig. 5.1) contains that the pin *in*, i.e., $s_2.drain$, must be connected to one of the gates of its subblocks ($\mathcal{R}_f : s_2.drain \leftrightarrow (s_1.gate \vee s_2.gate)$).

$S_1, \ldots, S_i$:	Structural implementation sets of $\text{FB}_1, \ldots, \text{FB}_i$ ($2, \ldots, i$ optional)
R_c (optional):	Characteristic connections of s_{new}
$\mathcal{R}_f$ (optional):	Rules s_{new} must fulfill
R_a (optional):	Additional connections s_{new} can have
$P_{\text{FB}_{new}}$:	Pin set of s_{new}

Figure 5.2.: Input of Algorithm 5, FB_j: jth functional block, s_j: a structural implementation of FB_j

Algorithm 5 Synthesis of a functional block except op-amp bias

Require: Compare Fig. 5.2

```
1: S_new := { } // The set of structural implementations of FB_new is empty
2: for all s_1 ∈ S_1 do
3:   for all s_2 ∈ S_2 do
4:     ....
5:     for all s_i ∈ S_i do
6:       c_new := createConnections(s_1, s_2, ..., s_i, R_c)
7:       if fulfillesRules(R_f,c_new) then
8:         s_new := createNewInstance(c_new, P_fb_new)
9:         S_new := S_new ∪ {s_new}
10:      end if
11:      for all r_a ∈ R_a do
12:        c_new := createConnections(c_new, r_a)
13:        if fulfillesRules(R_f,c_new) then
14:          s_new := createNewInstance(c_new, P_fb_new)
15:          S_new := S_new ∪ {s_new}
16:        end if
17:      end for
18:      ...
19:    end for
20:  end for
21: end for
22: return S_new
```

$\mathcal{R}_f$ also contains *basic structural rules* of analog building blocks as, e.g., that no transistor drain $t_m.drain$ is allowed to be connected to another transistor drain $t_n.drain$ of the same doping Φ:

$$\forall_{t_m,t_n \in T_\Phi} t_m.drain \nleftrightarrow t_n.drain \tag{5.1}$$

T_Φ are all transistors in the newly created implementation s_{new} of FB_{new} with doping type Φ. If the diode pair (*dip*) in Fig. 5.1 would have a connection between out_1 and out_2 it would not be a valid structural implementation of a voltage bias.

Additional connections R_a formulate additional optional connections to the connections in R_c. In the $vr1$-implementation of a cascode voltage bias (Fig. 5.1), the gates of the transistor are additionally connected. In the voltage reference 2, the gate of the lower transistor is additionally connected to the drain of the transistor above ($R_a : \{s_1.gate \leftrightarrow s_2.gate\}, \{s_2.drain \leftrightarrow s_1.gate\}$).

Algorithm

To synthesize all structural implementations of a functional block, Alg. 5 iterates over the sets of structural implementations $S_1, S_2, .., S_i$ to create all possible combinations. For each combination, a subcircuit c_{new} is created, consisting of $s_1 \in S_1$, $s_2 \in S_2$, ..., $s_i \in S_i$ having the required connections in R_c. It is checked if c_{new} fulfills the rules in $\mathcal{R}_f$. If that is the case, a new instance s_{new} is created being a valid structural implementation of FB_{new}. To

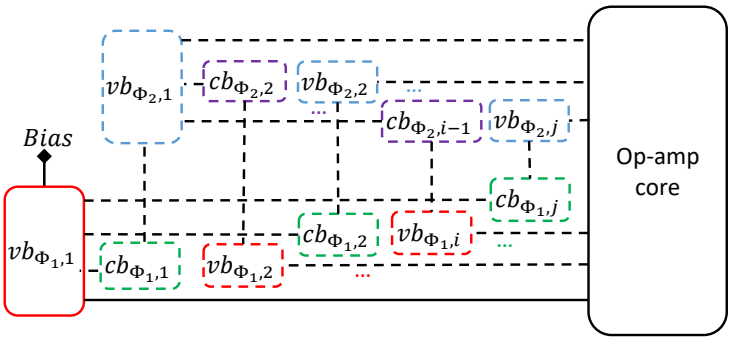


Figure 5.3.: Schematic overview of a bias

create all implementations of a cascode voltage bias, every combination of diode and normal transistor with the same doping having a drain-source connection is created. Because of R_f, only the diode pair and the mixed pair are recognized as valid structural implementations.

If a set R_a is provided, subcircuits having the defined additional connections are also created. If these circuits fulfill $\mathcal{R}_f$, the respective instances are created. Voltage reference 1 and voltage reference 2 are thus created as cascode voltage bias. Further transistor structures are created based on R_a, but are discarded as they do not fulfill $\mathcal{R}_f$.

5.2. Synthesis of the Op-Amp Bias[2]

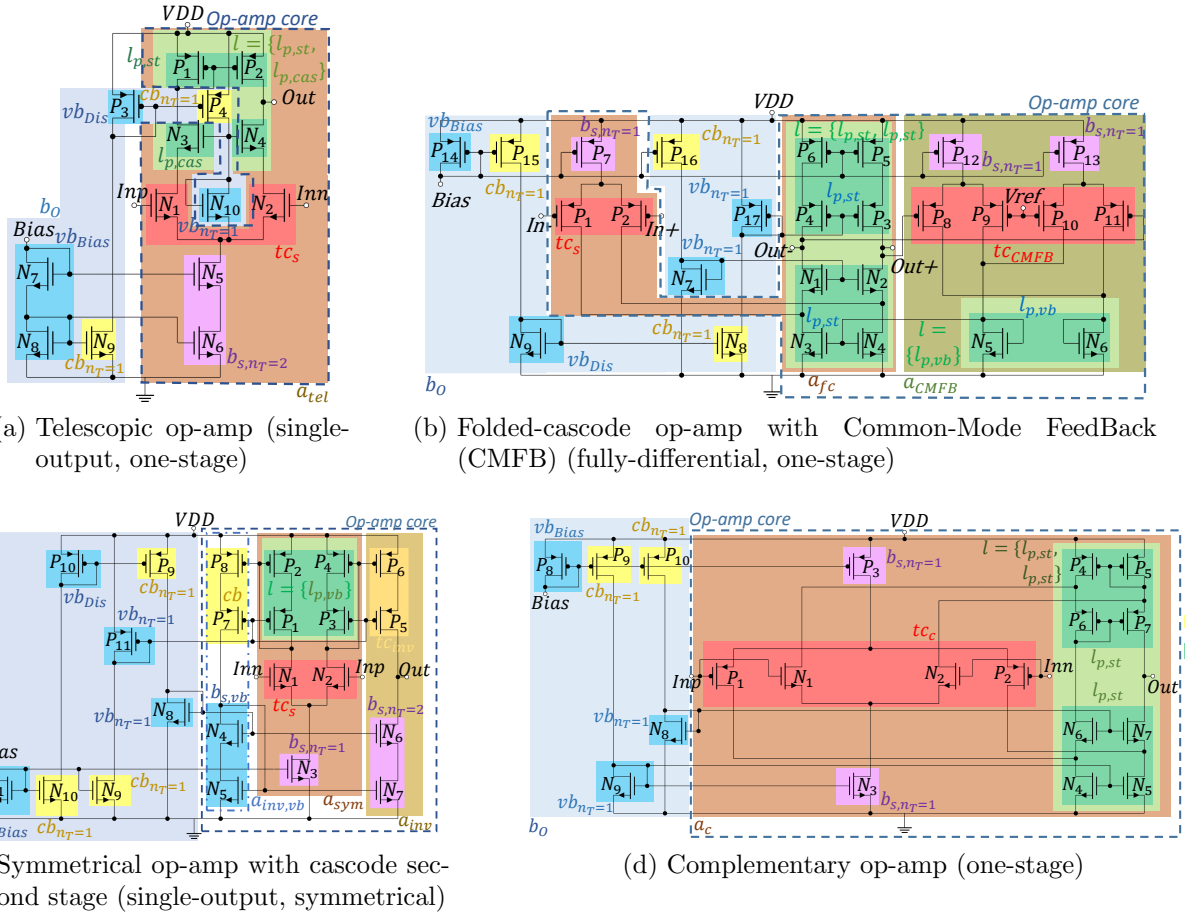
While Alg. 5 is used to synthesize the *op-amp core*, i.e., the amplification stages and capacitor structures with their connections (3.7) - (3.36), (3.38), (3.39), the op-amp bias b_O (3.37) is created with Alg. 6.

5.2.1. Structure of the Op-Amp Bias

An op-amp bias b_O consists of n voltage biases (vb) and $n-1$ current biases (cb) (Chapter 3). Its generic structure is defined in Fig. 5.3. The voltage biases supply the op-amp core with the required voltage potentials. The current biases supply these voltage biases with a current. One voltage bias of each doping ($vb_{\Phi_1,1}$, $vb_{\Phi_2,1}$) supplies these current biases, called *distributor voltage bias* (vb_{Dis}) in the following. A single current input pin p_{Bias} remains for the user-specified bias input current. The number of voltage biases forming the bias depends on the position of the transistors in the op-amp core needing voltage supply. Four different position types are distinguished:

1. *Improved Wilson current biases* are cascode current biases which have a diode transistor at the source and a normal transistor at the output. This type of current bias can only be connected to one implementation of a voltage bias (Fig. 5.1 $mp1$). Together, they form an improved Wilson current mirror (Fig. 5.4d $P_4 - P_7$). For each Wilson current bias in an op-amp, the specific voltage bias must be created (Alg. 6, Line 3).

[2]A similar version of this section was published in [62], reprinted with permission from "Inga Abel, Helmut Graeb, FUBOCO: Structure Synthesis of Basic Op-Amps by FUnctional BlOck COmposition, ACM Trans. Des. Autom. Electron. Syst." © 2022 ACM.



(a) Telescopic op-amp (single-output, one-stage)

(b) Folded-cascode op-amp with Common-Mode FeedBack (CMFB) (fully-differential, one-stage)

(c) Symmetrical op-amp with cascode second stage (single-output, symmetrical)

(d) Complementary op-amp (one-stage)

Figure 5.4.: Example topologies synthesizable with the presented method, colored background: functional blocks of HL 2 - 4

2. *Cascode current biases* can be biased by a cascode voltage bias (Fig. 5.4a, $N_1 - N_4, N_7, N_8$).
3. *Simple current biases with the source connected to the supply-voltage rail* are biased by simple voltage biases (Fig. 5.4b, $P_7 - P_{14}$)
4. *Simple current biases not connected to a supply-voltage rail* must be biased by an additional simple voltage bias. This is, e.g., the case in wide-swing cascode current mirrors (Fig. 5.4c, $N_4 - N_7$)).

A cascode current bias with doping Φ is not always biased by a cascode voltage bias. If additional single transistors of doping Φ are in the circuit needing voltage supply, the cascode current bias might be supplied by two simple voltage biases (Fig. 5.4d, $N_3 - N_9$).

5.2.2. Generic Algorithm to Synthesize the Op-Amp Bias b_O

Input of Alg. 6 are the transistors of the op-amp core needing voltage supply T_{un} and the functional blocks of the op-amp core $op = \{a_1, ...\}$. The transistors in T_{un} are sorted according to their doping Φ_1, Φ_2 and their position in the op-amp core. $T_{un,1}$ contains transistors connected with their source to a supply voltage rail, $T_{un,2}$ the remaining transistors. To create the bias b_O of the fully-differential op-amp (Fig. 5.4b), $T_{un,p,1} = \{P_5, P_6, P_7, P_{12}, P_{13}\}$, $T_{un,p,2} = \{P_3, P_4\}$, $T_{un,n,1} = \{\ \}$, $T_{un,n,2} = \{N_1, N_2\}$.

Algorithm 6 Synthesis of op-amp bias

Require: Set of transistors without voltage supply sorted according their doping and position in the op-amp core $T_{un} := T_{un,1,\Phi_1} \cup T_{un,2,\Phi_1} \cup T_{un,1,\Phi_2} \cup T_{un,2,\Phi_2}$; Set of functional blocks forming the op-amp-core $op = \{a_1, ...\}$

1: $VB_{\Phi_1} := \{\ \}$ //Set of voltage biases of doping Φ_1
2: $VB_{\Phi_2} := \{\ \}$ //Set of voltage biases of doping Φ_2
3: $VB_{\Phi_1,iw}, VB_{\Phi_2,iw}$:=createImprovedWilsonVoltageBiases(T_{un})
4: $VB_{\Phi_1,add}$:= createAdditionalVoltageBiases($T_{un,1,\Phi_1}$, $T_{un,2,\Phi_1}$) //Alg. 7
5: $VB_{\Phi_1} := VB_{\Phi_1} \cup VB_{\Phi_1,iw} \cup VB_{\Phi_1,add}$
6: $VB_{\Phi_2,add}$:= createAdditionalVoltageBiases($T_{un,1,\Phi_2}$, $T_{un,2,\Phi_2}$) //Alg. 7
7: $VB_{\Phi_2} := VB_{\Phi_2} \cup VB_{\Phi_2,iw} \cup VB_{\Phi_2,add}$
8: setBiasPin(VB_{Φ_1}, VB_{Φ_2})
9: CB, VB_{dis} := createCurrentBiases(VB_{Φ_1}, VB_{Φ_2})//Alg. 8
10: $VB := VB_{\Phi_1} \cup VB_{\Phi_2} \cup VB_{dis}$
11: **return** CB, VB

Algorithm 7 Create additional voltage biases

Require: Set of transistors without voltage supply sorted according their doping and position in the op-amp core $T_{un,1,\Phi}$, $T_{un,2,\Phi}$

1: $VB_\Phi := \{\ \}$ //Set of voltage biases of doping Φ
2: **if** $T_{un,1,\Phi} \cup T_{un,2,\Phi} = \{cb_{1,n_T=2}, cb_{2,n_T=2}, ...\}$ **then**
3: vb_Φ := createCascodeVoltageBias($T_{un,1,\Phi}$, $T_{un,2,\Phi}$)
4: $VB_\Phi := VB_\Phi \cup \{vb_\Phi\}$
5: **else**
6: $vb_{1,\Phi}$:= createSimpleVoltageBias($T_{un,1,\Phi}$)
7: $vb_{2,\Phi}$:= createSimpleVoltageBias($T_{un,2,\Phi}$)
8: $VB_\Phi := VB_\Phi \cup \{vb_{1,\Phi}\} \cup \{vb_{2,\Phi}\}$
9: **end if**
10: **return** VB_Φ

The algorithm creates and connects the improved Wilson voltage biases for all Wilson current biases in the op-amp core (Alg. 6, Line 3). For the folded-cascode op-amp (Fig. 5.4b), no Wilson voltage bias is created as there is no Wilson current bias in the circuit.

To supply the remaining cascode and simple current biases, voltage biases are created with Alg. 7. If all remaining transistors of doping Φ in $T_{in,\Phi}$ are part of cascode current biases, a cascode voltage bias is created and connected to supply these transistors (Line 3). Otherwise, simple voltage biases are created to connect the transistors in the two sets $T_{un,1,\Phi}$, $T_{un,2,\Phi}$ (Line 6, 7). If one of the sets is empty, the corresponding voltage bias is not created. For the folded-cascode op-amp (Fig. 5.4b), three voltage biases are created: P_{14} for $T_{un,p,1}$, P_{17} for $T_{un,p,2}$, N_7 for $T_{un,n,2}$. Although P_3, P_5 and P_4, P_6 form cascode current biases, as $T_{un,p,1}$ contains additional transistors not being part of a cascode current bias (P_7, P_{12}, P_{13}), no cascode voltage bias is created.

From the created voltage biases, one voltage bias is chosen to be connected to the current bias input pin p_{Bias} (Alg. 6, Line 8). In the selection process, a single voltage bias is preferred to a cascode one, which is preferred to an improved Wilson voltage bias. This voltage bias vb_{Bias} is already set to be a distributor voltage bias for the later created current biases

Algorithm 8 Create current biases

Require: Set of voltage biases being of one doping with the bias pin included $VB_{\Phi_{Bias}}$; Set of voltage biases being of the other doping $VB_{\Phi_{Other}}$

1: $CB := \{\ \}$ //Set of current biases
2: $VB_{Dis} := \{\ \}$ //Set of distributor voltage biases
3: **if** $|VB_{\Phi_{Bias}}| > 1$ **then**
4: vb_{Dis} := findDistributorVoltageBias($VB_{\Phi_{Other}}$)
5: $CB_{\Phi_{Other}}$:= createCurrentBiases(vb_{Dis}, $VB_{\Phi_{Bias}}$)
6: $CB := CB \cup CB_{\Phi_{Other}}$
7: **if** $VB_{\Phi_{Other}} \cap vb_{Dis} = \{\ \}$ **then**
8: $VB_{Dis} := VB_{Dis} \cup vb_{Dis}$
9: **end if**
10: **end if**
11: **if** $VB_{\Phi_{Other}} \neq \emptyset$ **then**
12: $CB_{\Phi_{Bias}}$:= createCurrentBiases(vb_{Bias}, $VB_{\Phi_{Other}}$)
13: $CB := CB \cup CB_{\Phi_{Bias}}$
14: **end if**
15: **return** CB, VB_{Dis}

of the same doping (Fig. 5.3). For the folded-cascode op-amp (Fig. 5.4b), P_{14} is chosen as vb_{Bias}. Different to P_{17}, N_7, it biases transistors which have a source connection to the supply-voltage rail.

Alg. 8 creates the current biases of the op-amp bias b_O. Its input are the sets of voltage biases ordered according to their doping $VB_{\Phi_{Bias}}, VB_{\Phi_{Other}}$. The algorithm first creates all current biases of doping Φ_{Other}. Current biases of this doping are only needed when the number of voltage biases having the doping Φ_{Bias} is larger than one. As the current biases must be connected to a *distributor voltage bias*, a respective voltage bias must be selected in $VB_{\Phi_{Other}}$ using the same criteria as for vb_{Bias}. If $VB_{\Phi_{Other}}$ is empty or contains only voltage biases supplying transistors in $T_{un,2}$ (Fig. 5.4b, N_7), a simple voltage bias is created to be the distributor voltage bias vb_{Dis} (Fig. 5.4b, N_9). If voltage biases of doping Φ_{Other} were created during the synthesis process, current biases of doping Φ_{Bias} are created. The distributor voltage bias for them is vb_{Bias}. For the folded-cascode op-amp (Fig. 5.4b), $VB_{\Phi_{Bias}} = \{P_{14}, P_{17}\}$, $VB_{\Phi_{Other}} = \{N_7\}$. P_{14} is connected to the bias pin. To bias P_{17}, the current bias N_8 is created and added to $VB_{\Phi_{Other}}$. As $VB_{\Phi_{Other}}$ does not have a voltage bias that biases transistors in $T_{un,1}$, P_9 is created as distributor voltage bias of Φ_{Other}. Two current biases P_{15}, P_{16} are created to bias $N_7, N_9 \in VB_{\Phi_{Other}}$.

5.3. Functional Block Composition (FUBOCO) of Complete Op-Amp Topologies[3]

The described algorithms (Algs. 5 - 8) are combined to synthesize complete op-amp topologies. Fig. 5.5 shows the composition rules to synthesize all implementations of functional

[3]A similar version of this section was published in [62], reprinted with permission from "Inga Abel, Helmut Graeb, FUBOCO: Structure Synthesis of Basic Op-Amps by FUnctional BlOck COmposition, ACM Trans. Des. Autom. Electron. Syst." © 2022 ACM.

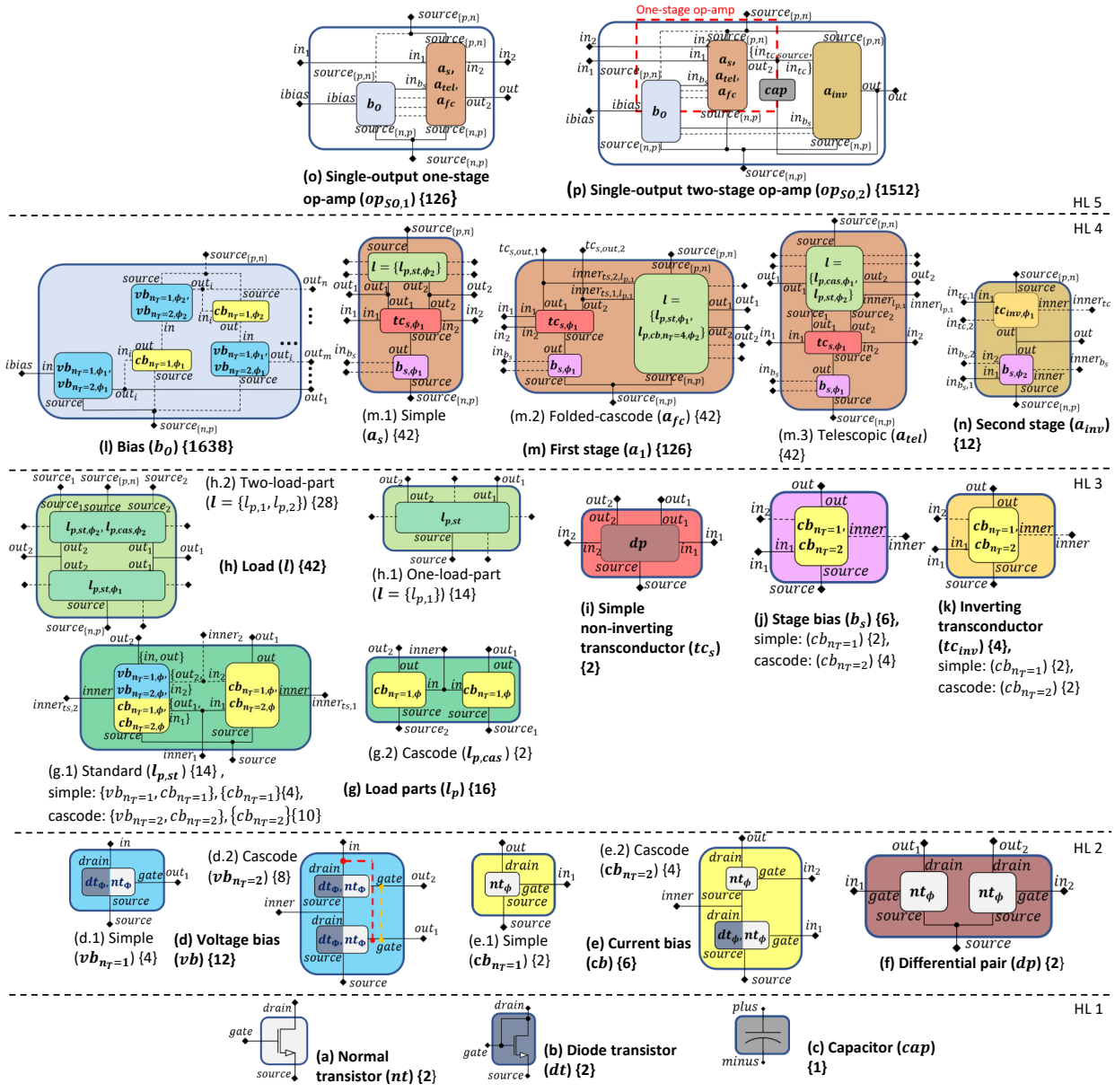


Figure 5.5.: Functional block composition rules for single-output op-amps. $\{n\}$ denotes the respective number of synthesizable structural implementations

blocks for single-output one-stage and two-stage op-amps (e.g., Fig. 5.4a). The composition rules for additional functional blocks for single-output symmetrical op-amps (e.g., Fig. 5.4c), fully-differential op-amps (e.g., Fig. 5.4b) and complementary op-amps (e.g., Fig. 5.4d) are given in Appendix B. Structural examples synthesizable for each functional block type are given in Chapter 3.

5.3.1. Hierarchy Level 1: Devices

For every device type, instances are created (Fig. 5.5a-c). For transistors, it is differentiated between n- and p-doping (Φ_n, Φ_p).

5.3.2. Hierarchy Level 2: Structures

Two types of *voltage bias (vb)* implementations (Fig. 5.5d) are synthesized by Alg. 5. For the *simple voltage bias* ($vb_{n_T=1}$, Fig. 5.5d.1), only one set of implementations is inputted into the algorithm consisting of normal and diode transistors ($S_1 : NT, DT$). The synthesis of *cascode voltage bias* implementations ($vb_{n_T=2}$, Fig. 5.5d.2) is discussed in Sec. 5.1.

Current biases (*cb*, Fig. 5.5e) are synthesized similar to the voltage biases. *Simple current biases* ($cb_{n_T=1}$, Fig. 5.5e.1) only consist of normal transistors ($S_1 : NT$, e.g., Fig. 5.4b P_{15}). The *cascode variant* ($cb_{n_T=2}$, Fig. 5.5e.2) consists either of a diode and a normal transistor (e.g., Fig. 5.4d P_5, P_7) or two normal transistors (e.g., Fig. 5.4c P_7, P_8) having a drain-source connection ($S_1 : NT_\Phi, DT_\Phi$; $S_2 : NT_\Phi$; $R_c : s_1.drain \leftrightarrow s_2.source$).

Differential pairs (*dp*, Fig. 5.5f) are created using two normal transistors (*nt*) of the same doping Φ connected at the sources (e.g., Fig. 5.4d, N_1, N_2) ($S_1 : NT_\Phi$; $S_2 : NT_\Phi$; $R_c : s_1.source \leftrightarrow s_2.source$;).

5.3.3. Hierarchy Level 3: Amplification Stage Subblocks

Two different types of *load parts (l_p)* are synthesized for single-output op-amps (Fig. 5.5g):

The *standard load part* ($l_{p,st}$, (Fig. 5.5g.1)) is synthesized either based on two current biases of the same implementation (e.g., Fig. 5.4d $N_4 - N_7$), or based on a voltage and a current bias (e.g., Fig. 5.4d $P_4 - P_7$). The voltage and current biases are either simple (e.g., Fig. 5.4a $P_1 - P_2$) or cascode (e.g., Fig. 5.4b $P_3 - P_6$). The functional subblocks s_1, s_2 are connected at their sources. Also the inputs are connected or, in cases of a load part based on a voltage and a current bias, the outputs are connected to the inputs.

$S_1 : CB_\Phi, VB_\Phi$; $S_2 : CB_\Phi$; $R_c : n_{T,fb_1} = n_{T,fb_2}$, $s_1.source \leftrightarrow s_2.source$,
$(s_1.in_1 \vee s_1.out_1) \leftrightarrow s_2.in_1$, $\forall_{n_{T,s_1}=2}(s_1.in_2 \vee s_1.out_2) \leftrightarrow s_2.in_2$;

Cascode load part implementations ($l_{p,cas}$, Fig. 5.5g.2) are only relevant for the synthesis of telescopic op-amps (Fig. 5.4a). The load part is synthesized based on simple current biases (*cb*) being only connected at the input pins (N_3, N_4) ($S_1 : CB_{n_T=1,\Phi}$; $S_2 : CB_{n_T=1,\Phi}$; $R_c : s_1.in_1 \leftrightarrow s_2.in_1$;).

Loads (l, Fig. 5.5h) are synthesized using either one (Fig. 5.4c, $P_1 - P_4$) or two load parts (Fig. 5.4b, $P_3 - P_6, N_1 - N_4$).

One-load-part loads implementations ($l = \{l_{p,1}\}$, Fig. 5.5h.1) are created with standard load parts ($S_1 : L_{p,st}$).

Two-load-part loads ($l = \{l_{p,1}, l_{p,2}\}$, Fig. 5.5h.2) consist of two standard load parts l_{p,st,Φ_1}, l_{p,st,Φ_2} with different doping (e.g., Fig. 5.4b $N_1 - N_4$, $P_3 - P_6$) or, iff the op-amp has a telescopic first stage, a standard load part l_{p,st,Φ_1} and a cascode load part l_{p,cas,Φ_2} (Fig. 5.4a N_3, N_4, P_1, P_2). They are connected at the load part outputs ($S_1 : L_{p,st,\Phi_1}$; $S_2 : L_{p,st,\Phi_2}$,L_{p,cas,Φ_2}; $R_c : s_1.out_1 \leftrightarrow s_2.out_1, s_1.out_2 \leftrightarrow s_2.out_2$;).

Simple non-inverting transconductors (tc_s, Fig. 5.5i) are synthesized based on one differential pair ($S_1 : DP$) (e.g., Fig. 5.4b P_1, P_2).

Stage biases (b_s, Fig. 5.5j) are created based on simple (Fig. 5.4b P_7) or cascode (Fig. 5.4a, N_5, N_6) current biases ($S_1 : CB$).

Inverting transconductors (tc_{inv}, Fig. 5.5k) are based on current biases ($S_1 : CB$;), which are simple or cascode. No connection between the first input pin and the inner pin is allowed ($\mathcal{R}_f : s_1.in_1 \nleftrightarrow s_1.inner$).

5.3.4. Hierarchy Level 4: Amplification Stages

The topology-specific *op-amp bias* b_O (Fig. 5.5l) is synthesized using Alg. 6 after the amplification stages and capacitors are created and connected. The bias consists of voltage and current biases (e.g., Fig. 5.4b $P_{14} - P_{17}$, $N_7 - N_9$).

Three different types of *first stage* implementations (a_1, Fig. 5.5m) are supported for simple op-amps:

Simple first stages (a_s, Fig. 5.5m.1) are synthesized based on a one-load-part load, a simple non-inverting transconductor and a stage bias ($S_1 : TC_{s,\Phi_1}$; $S_2 : B_{s,\Phi_1}$; $S_3 : L = \{L_{p,st}\}$). The load is of different doping Φ_2 than the transconductor and stage bias (Φ_1). The transconductor's source is connected to the output of the stage bias, while its outputs are connected to the outputs of the load ($R_c : s_1.source \leftrightarrow s_2.out$, $s_1.out_1 \leftrightarrow s_3.out_1$,$s_1.out_2 \leftrightarrow s_3.out_2$;).

Folded-cascode first stage implementations (a_{fc}, Fig. 5.5m.2) are synthesized with loads consisting of two standard load parts. One load part $l_{p,cb,n_T=4,\Phi_2}$ of the two-load-part load consists of current biases, has four transistors and a different doping than tc_s. This load part is connected with its inner pins of the current biases $inner_{ts_1,l_{p,1}}, inner_{ts_2,l_{p,1}}$ to the output pins of the transconductor.

$S_1 : TC_{s,\Phi_1}$; $S_2 : B_{s,\Phi_1}$; $S_3 : L = \{L_{p,st,\Phi_1}, L_{p,cb,n_T=4,\Phi_2}\}$; $R_c : s_1.source \leftrightarrow s_2.out$, $s_1.out_1 \leftrightarrow s_3.inner_{ts_1,l_{p,1}}$, $s_1.out_2 \leftrightarrow s_3.inner_{ts_2,l_{p,1}}$;

Telescopic first stages (a_{tel}, Fig. 5.5m.3) are created with a two-load part load consisting of a cascode and a standard load part $L = \{L_{p,cas,\Phi_1}, L_{p,st,\Phi_2}\}$.

$S_1 : TC_{s,\Phi_1}$; $S_2 : B_{s,\Phi_1}$; $S_3 : L = \{L_{p,cas,\Phi_1}$,$L_{p,st,\Phi_2}\}$; $R_c : s_1.source \leftrightarrow s_2.out$, $s_1.out_1 \leftrightarrow s_3.source_1$, $s_1.out_2 \leftrightarrow s_3.source_2$;

Second stages (a_{inv}, Fig. 5.5n) are composed of an inverting transconductor tc_{inv} and a stage bias b_s of different doping ($S_1 : TC_{inv,\Phi_1}$; $S_2 : B_{s,\Phi_2}$). They are connected at their outputs ($R_c : s_1.out \leftrightarrow s_2.out$,).

5.3.5. Hierarchy Level 5: Op-Amps

Two types of *single-output op-amps* (op_{SO}) are supported:

One-stage op-amp implementations ($op_{SO,1}$, Fig. 5.5n) are created by synthesizing different first stage implementations ($S_1 : A_s, A_{tel}, A_{fc}$) with Alg. 5. The bias circuit b_O is synthesized with Alg. 6.

Two-stage op-amps ($op_{SO,2}$, Fig. 5.5p) are created by adding a second stage a_{inv} and a capacitor *cap* to the first stage of the one-stage op-amp using Alg. 5 ($S_1 : A_s, A_{tel}, A_{fc}$; $S_2 : cap$; $S_3 : A_{inv}$;). The capacitor is connected between the output of the first stage and the second stage, the output of the first stage is connected to the input of the second stage ($R_c : s_1.out_2 \leftrightarrow s_2.plus$, $s_1.out_2 \leftrightarrow s_3.in_{tc,1}$, $s_2.minus \leftrightarrow s_3.out$;). The bias b_O is synthesized with Alg. 6.

5.3.6. Number of Implementations per Functional Block

Fig. 5.5 gives the maximum number of structural implementations per functional block currently supported by the synthesis algorithm. It includes n-type as well as p-type implementations and can be controlled by adding/removing rules to the set of functional block rules R_f and adding/removing functional block implementation from S_i. E.g., removing the diode transistors from S_1 of the cascode current bias leads to only two implementations of the cascode current bias instead of four, reducing also the number of synthesized stage biases and thus also the number of first and second stages as well as overall op-amp topologies.

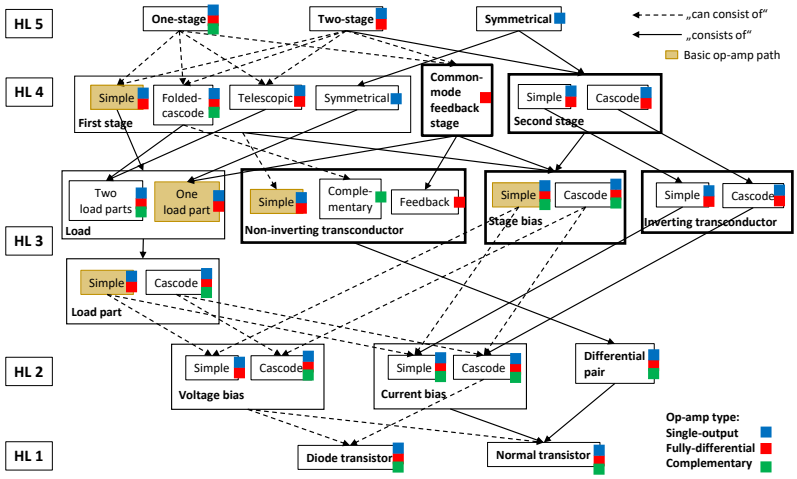
5.4. Overview of the Complete FUBOCO Synthesis Process[4]

Fig. 5.6 gives an overview of the overall synthesis algorithm. The parts of the hierarchical functional block composition graph (Fig. 5.6a) marked with blue dots represent an abstraction of the composition graph for single-output op-amps (SO) in Fig. 5.5. The parts marked with red and green dots represent the functional blocks needed to synthesize fully-differential (FD, red) and complementary op-amps (Comp, green). Details of the supplementary functional blocks are given in Appendix B. The op-amp type which has to be synthesized and its performance requirements F_B are specified by the user.

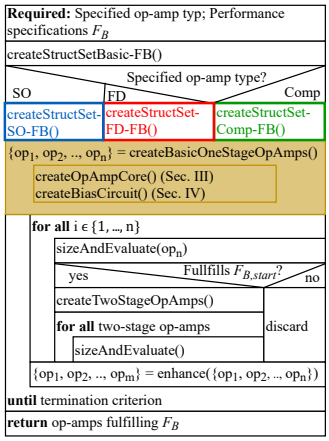
5.4.1. Functional Block Composition (FUBOCO) Graph

The composition graph (Fig. 5.6a) defines how to compose each functional block on level x from a set of functional blocks on level x-1 or x by combination, starting from the functional blocks on the 5th hierarchy level (HL), i.e., one-stage, two-stage, or symmetrical op-amp. The usage of a functional block is either strict ("consists of"), or a selection of one out of many ("can consist of"). The new composition graph differs from the design plan-based structural synthesis approaches [36,38], which feature a single solution path through such a composition

[4] A similar version of this section was published in [62], reprinted with permission from "Inga Abel, Helmut Graeb, FUBOCO: Structure Synthesis of Basic Op-Amps by FUnctional BlOck COmposition, ACM Trans. Des. Autom. Electron. Syst." © 2022 ACM.



(a) Functional block composition graph (without op-amp bias). HL 1 - HL 2: Basic functional blocks, HL 3 - HL 4, thick frame: Basic functional blocks of a specific op-amp type.



(b) Structure synthesis algorithm.

Figure 5.6.: Overview of the FUBOCO synthesis process.

graph based on if-then-else decisions, and from the structural synthesis approaches that use local structural changes with a nearly open-end process. The FUBOCO graph instead provides a large search space of yet only technically meaningful structures and uses a fast equation-based sizing process for an optimization-based selection process.

Each op-amp type features its own part of the functional block composition graph for synthesis. The respective blocks are marked with the respective color in Fig. 5.6a. The Common-Mode FeedBack stage (CMFB stage) for instance is only used in fully-differential op-amps (red), a folded cascode first stage is used for all three types of op-amps (blue, red, green). One-stage, two-stage and symmetrical types of single-output op-amps are considered, one- and two-stage versions of fully-differential op-amps can be synthesized, and one-stage complementary op-amps are supported.

5.4.2. Synthesis Algorithm

The synthesis algorithm, sketched in Fig. 5.6b, features a combination of enumerative and generative approaches to structure synthesis. The evaluation of created structural op-amp variants is based on optimization over behavorial equations and is particularly fast.

Three groups of functional blocks are distinguished and specifically treated in the structural synthesis process:

Basic functional blocks are all functional blocks of HL 1 - HL 2 (Fig. 5.6a). They are part of many other functional blocks, e.g., non-inverting and inverting transconductor, load and stage bias. The number of implementations per functional block is small (2 - 12 for each type of functional block).

Basic functional blocks of a specific op-amp type are the functional blocks of HL 3 - HL 4 framed with thick lines. Not all functional blocks are part of every op-amp type, such that the functional blocks can be further divided into op-amp type specific groups having overlaps. The stage bias, for instance, is part of every group, the common-mode feedback stage is only

relevant for fully-differential op-amps. The number of implementations per functional blocks is small (2 - 12).

Functional blocks with many implementations are the op-amps themselves, the first stages, loads and load parts. The number of implementations per functional block is high and varies between 24 for the load parts and 318 for the first stages.

Based on the methods presented in the preceding sections, the algorithm creates all implementations of basic functional blocks and of op-amp type-specific functional blocks upfront and stores them in a library (parts marked with blue, red, green dots in Fig. 5.6b). Functional blocks with a large number of structural implementation are only created on-demand when they are part of a topology. This provides a good compromise between computation time and memory usage.

The algorithm creates a set of basic one-stage op-amps with a low number of transistors. This set is marked with golden background in Fig. 5.6a and refers to the part similarly colored in the synthesis algorithm in Fig. 5.6b. The topologies are evaluated based on sizing (Sec. 5.4.3). Some op-amp characteristics F_{start} degenerate or do not change if a second stage is added to a one-stage op-amp. If a one-stage op-amp does not fulfill the corresponding specifications $F_{B,start}$, its two-stage versions would also not fulfill the specifications. Hence, two-stage variants are only created if these performance specifications are satisfied. Otherwise, a one-stage op-amp variant and its potential two-stage variants are discarded, thus bounding and reducing the search tree from irrelevant branches.

One-stage op-amps are enhanced by changing their stage bias and/or load. A new set of one-stage op-amps and their two-stage variants is configured and evaluated. The synthesis process ends if either the simplest op-amp fulfilling the specifications is found or all op-amp topologies are enumerated.

Please note that complementary op-amps and symmetrical op-amps are synthesized slightly different. For complementary op-amps, only one-stage op-amps are currently considered. Symmetrical op-amps exist only as two-stage variants, such that the differentiation between one- and two-stage variant is not made.

5.4.3. Topology Sizing and Evaluation

For topology sizing and evaluation, the sizing method in Chapter 4 is used. The optimization approach within the initial sizing method is enhanced for synthesis. Two groups of performance variables are defined. The group F_{start} contains all performance variables which depend only on design variables of the first stage or degrade by adding a second stage to the one-stage op-amp. These are the transistor gate-area f_D, the power consumption f_P, the maximum and minimum common-mode input voltage $f_{v_{cm,max}}, f_{v_{cm,min}}$, the common-mode rejection ratio f_{CMRR} and the phase margin f_{PM}.

$$F_{start} = \{f_D, f_P, f_{v_{cm,max}}, f_{v_{cm,min}}, f_{CMRR}, f_{PM}\} \tag{5.2}$$

The group F_{end} contains the remaining performance variables. These are the open-loop gain $f_{A_{D,0}}$, the slew rate f_{SR}, the unity-gain bandwidth f_{GBW} and the maximum and minimum output voltage swing $f_{v_{out,max}}, f_{v_{out,min}}$:

$$F_{end} = \{f_{A_{D,0}}, f_{SR}, f_{GBW}, f_{v_{out,max}}, f_{v_{out,min}}\} \tag{5.3}$$

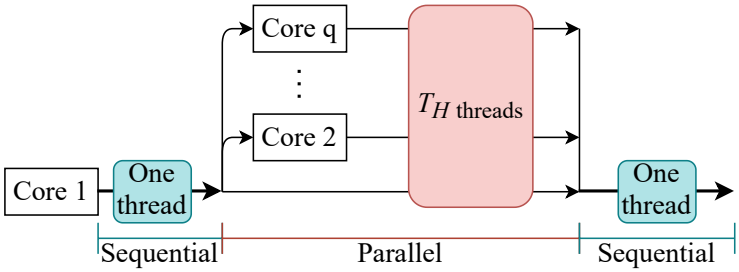


Figure 5.7.: Sequential and parallel tasks

The performance variables in the set F_{start} must fulfill the user-given constraint values $F_{B,start}$. Otherwise, the topology is discarded from the synthesis process. If it was a one-stage op-amp, its two-stage variants are not considered in the synthesis process.

The variables of F_{end} are optimized towards over-fulfillment of their specifications $F_{B,end}$. If this succeeds, all performance variables $F = F_{start} \cup F_{end}$ are further optimized. In this work, the optimization algorithm is based on Constraint Programming and does not seek to reach an optimum but is terminated when the optimization progress slows down (Chapter 4). Experiments have shown that after an optimization time of around one minute per structural variant the progress slows down. We therefore set the time limit for one optimization run to one minute.

The analytical equation-based sizing method allows the emulation of the manual sizing process during topology evaluation, this is different to other approaches, e.g., [40, 45], using numerical, simulation-based sizing methods as, e.g., [17, 72, 98].

5.5. Runtime Reduction Based on Multi-Threading Strategies[5]

The sizing and evaluation process takes one minute per topology (Sec. 5.4.3). As the synthesis process supports several thousands of topologies, this may lead to long runtimes. To reduce the runtime, we use multi-threading strategies to parallelize the sizing and evaluation process in the synthesis algorithm such that multiple op-amp topologies are sized and evaluated at the same time. This reduces the runtime of the synthesis algorithm by approximately 80% depending on the used hardware.

In the following section, multi-threading is briefly described (Sec. 5.5.1). The changes on the synthesis algorithm are given in Sec. 5.5.2.

5.5.1. Multi-Threading

Many programming languages as C++ or Python allow to distribute tasks to multiple threads which run in parallel during program execution (Fig. 5.7). This is called multi-threading. Depending on the hardware, T_H threads can be executed in parallel. T_H does not have to be identical to the number of cores q of the used CPU as there exist cores which

[5]The content of this section was originally published in [63], reprinted with permission from "Inga Abel, Clara Kowalsky, Helmut Graeb, A fast Structural Synthesis Algorithm for Op-Amps based on Multi-Threading Strategies, International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)" © 2021 IEEE.

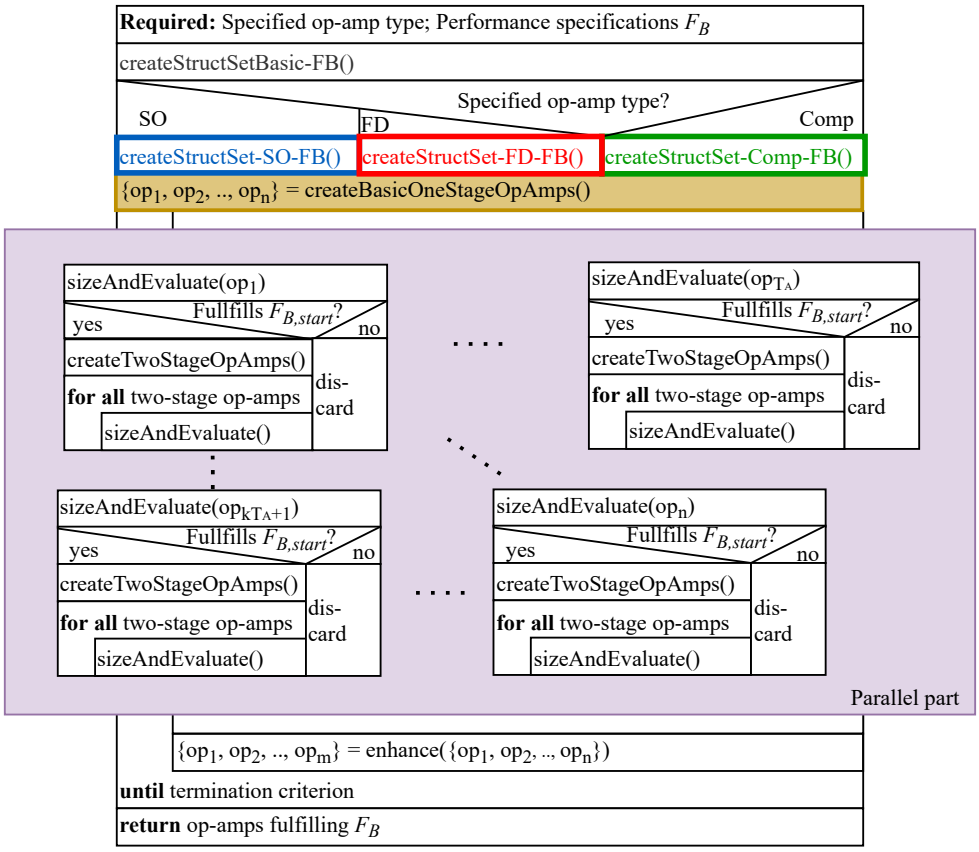


Figure 5.8.: Structural synthesis algorithm

can run multiple threads in parallel. Running tasks in parallel can result in a significant speed-up.

Not all tasks in a program are suited to run in parallel. The tasks running in parallel should be as independent as possible, needing only read access to common variables. Write access to common variables might lead to race conditions. A variable might exceed its limit, as two tasks write on it in the same time. To avoid race conditions, locks are used which allow a thread to claim a variable when it is writing on it.

In structural synthesis, the sizing of different circuits is a task suitable for parallel programming, as the sizing of one circuit is independent of the sizing of another circuit.

5.5.2. Structural Synthesis Algorithm with Parallelized Topology Evaluation

Different to Sec. 5.4, the sizing and evaluation of a set of topologies runs in parallel. Depending on the number of created op-amps, T_A - 1 threads are started in addition to the main thread in which the program runs. T_A depends on the number of threads the used hardware supports, T_H, and the number of created op-amps n. It is calculated by:

$$T_A = \begin{cases} T_H, \ if \ n \geq T_H \\ n, \ if \ n < T_H \end{cases} \tag{5.4}$$

Table 5.1.: Specifications; Specs 1, Specs 4, Specs 6: general specifications; Specs 2, Specs 3, Specs 5, Specs 7: sophisticated specifications

Specs #	1	2	3	4	5	6	7
Op-amp type	Single-output			Fully-differential		Complementary	
Bias current (μA)	10	100	10	100	100	100	100
Load capacity (pF)	20	20	20	20	20	20	20
Supply voltage (V)	5	5	5	5	5	5	5
Gate-area ($10^3\mu m^2$)	≤ 15	≤ 5	≤ 5	≤ 50	≤ 20	≤ 15	≤ 5
Quiescent power (mW)	≤ 15	≤ 8	≤ 5	≤ 25	≤ 15	≤ 10	≤ 5
Phase margin (°)	≥ 60	≥ 60	≥ 80	≥ 60	≥ 60	≥ 60	≥ 60
CMRR (dB)	≥ 70	≥ 70	≥ 70	≥ 80	≥ 80	≥ 80	≥ 80
CMIR (V)	2-3	1.5-3.5	1.5-3.5	2-3	2-3	-	-
Open-loop gain (dB)	≥ 80	≥ 70	≥ 45	≥ 70	≥ 60	≥ 70	≥ 70
Unity-gain bandwidth (MHz)	≥ 2.5	≥ 10	≥ 10	≥ 2.5	≥ 10	≥ 2.5	≥ 10
Slew rate ($\frac{V}{\mu s}$)	≥ 3.5	≥ 20	≥ 20	≥ 3.5	≥ 15	≥ 3.5	≥ 15
Output voltage swing (V)	1.5-3.5	1.5-3.5	1-4	2-3	1.5-3.5	1.5-3.5	1-4

Each thread t_i, $i \in \{1, .., T_A\}$ is assigned a set of k_i op-amps to be sized and evaluated. The number of topologies k_i depends on the number of created threads T_A and the number of created op-amp topologies n:

$$k_i = \begin{cases} \lfloor n/T_A \rfloor, \ for\ t_i, i \in \{1, .., T_A - (n \mod T_A)\} \\ \lceil n/T_A \rceil, \ for\ t_i, i \in \{T_A - (n \mod T_A) + 1, .., T_A\} \end{cases} \tag{5.5}$$

The main thread t_1 also runs the sequential parts of the program and therefore has a smaller number of assigned circuits if the circuits cannot be equally distributed. The thread t_1 also controls the starting and closing of the threads $t_2, ..., t_{T_A}$ created for the parallel execution of the program.

The sizing and evaluation of two-stage op-amps still runs sequentially, which lessens the effect of the inherent reduction of the search space when two-stage op-amps are not created if the one-stage op-amp did not fulfill $F_{B,start}$. Some threads running in parallel might have more one-stage op-amps for which two-stage op-amps are created than others. Future work remains in parallelizing the sizing and evaluation of two-stage op-amps to distribute tasks more equally.

5.6. Experimental Results

The synthesis tool was tested with seven different specification sets (Table 5.1). FUBOCO created for each set all topologies that fulfill the specifications. All supported implementations of every functional block were allowed in the synthesis process (Sec. 5.3.6).

Specs 1, Specs 2 and *Specs 3* (Table 5.1) specify single-output as op-amp type. *Specs 1* has a high gain requirement likely to exclude many topologies. *Specs 2* demands a smaller quiescent power and gate-area. Also, the requirements for slew rate and unity-gain bandwidth are more challenging. *Specs 3* requires an even smaller quiescent power. The requirements for the gain are less strict. A high phase margin is required. *Specs 4* and *Specs 5* are specified for fully-differential op-amps. In *Specs 4*, the most demanding specification is the gain being comparatively high. *Specs 5* demands a smaller area and quiescent power and a higher unity gain-bandwidth and slew rate. The op-amp type in *Specs 6* and *Specs 7* is complementary. *Specs 6* is satisfiable by many topologies variants of complementary op-amps. In *Specs 7*, the

Table 5.2.: Number of created topologies by FUBOCO; brackets: max. # supported topologies

Specs #	1	2	3	4	5	6	7
# one-stage op-amps fulfilling $F_{B,start}$	144 (210)	96 (210)	81 (210)	39 (72)	37 (72)	36 (36)	36 (36)
# created op-amp topologies	1728 (2940)	1152 (2940)	972 (2940)	468 (936)	444 (936)	36 (36)	36 (36)
# op-amps fulfilling F_B	228 (2940)	71 (2940)	54 (2940)	34 (936)	2 (936)	10 (36)	6 (36)

allowed area and quiescent power is reduced. The requirements for unity-gain bandwidth and slew rate increase.

Please note that the user specifies the op-amp type (single-output, fully-differential, complementary) and the performance requirements in the specification set. Additional design knowledge is not required. Also note that only the gate-area of every transistor $t \in T$, i.e. $\sum_{t \in T} W_t L_t$, is considered as area constraint. However, also other models including more layout aspects can be used in the calculation.

In the following section, the structures of the synthesized op-amp topologies are discussed (Sec. 5.6.1). Sec. 5.6.2 discusses the runtime of the synthesis algorithm. First, the general runtime behavior of the algorithm for different specification sets is discussed. A comparison of the runtime of the sequential algorithm in Sec. 5.4 with the parallel algorithm in Sec. 5.5.2 follows. Sizing and evaluation results for the different specifications (Table 5.1) are discussed in Sec. 5.6.3.

5.6.1. Synthesized Topologies[6]

Table 5.2 shows the number of topologies created by FUBOCO for the different specifications. FUBOCO currently supports 2940 single-output topologies of which 210 are one-stage topologies, 936 fully-differential topologies (72 one-stage/864 two-stage) and 36 complementary op-amps. Not all topologies are created in each run, as the syntheses process does not consider two-stage op-amps if its one-stage variant fails the specifications in $F_{B,start}$ (Sec. 5.4.2).

Table 5.3 gives an overview of the composition of amplification stages in the synthesized topologies for *Specs 1* - *Specs 5*. Topologies with a large structural variety are outputted for general specifications (*Specs 1*, *Specs 4*), a smaller variety results for more specific specifications (*Specs 2*, *Specs 3*, *Specs 5*).

For *Specs 1*, the largest number of op-amps were created. Many one-stage op-amps fulfilled $F_{B,start}$ (144), such that in total 1728 op-amps were created. 228 of the 1728 topologies fulfilled *Specs 1*. Due to the high gain requirement, most of the topologies are two-stage op-amps with a simple or a folded-cascode first stage (Table 5.3). The one-stage op-amps are either formed with a folded-cascode or telescopic first stage. The 60 symmetrical op-amps all have a cascode second stage. An example of an op-amp fulfilling the specifications is the symmetrical op-amp in Fig. 5.4c.

As *Specs 2* is more strict, the number of topologies fulfilling the set is much smaller: 96 one-stage op-amps fulfill $F_{B,start}$ leading to 1152 created op-amps in total. 92 of the 1152

[6]A similar version of this section was published in [62], reprinted with permission from "Inga Abel, Helmut Graeb, FUBOCO: Structure Synthesis of Basic Op-Amps by FUnctional BlOck COmposition, ACM Trans. Des. Autom. Electron. Syst." © 2022 ACM.

Table 5.3.: Amplification stage composition of the resulting topologies

First stage type	simple a_s		folded-cascode a_{fc}		telescopic a_{tel}		symmetrical a_{sym}	total #
# stages	1	2	1	2	1	2	-	topologies
Specs 1	0	40	30	68	30	0	60	228
Specs 2	0	5	18	0	20	0	28	71
Specs 3	24	0	0	0	3	0	27	54
Specs 4	0	0	11	22	1	0	-	34
Specs 5	0	0	2	0	0	0	-	2

topologies fulfill all specifications. The set is dominated by telescopic and folded-cascode one-stage op-amps and symmetrical op-amps (Table 5.3). Due to the strong area constraint, folded-cascode two-stage op-amps do not longer fulfill the specifications. The small number of two-stage op-amps all have a simple first stage. As the constraints on the input voltage are more demanding, all topologies in the set have a simple voltage bias as stage bias of the first stage. The symmetrical op-amp (Fig. 5.4c) is also a valid topology for *Specs 2*.

The set of topologies fulfilling *Specs 3* is even smaller. As 81 one-stage op-amps fulfilled the specifications in $F_{B,start}$, 972 op-amps in total were created. 54 topologies fulfilled all specifications F_B. Only one-stage op-amps and symmetrical op-amps fulfill the specifications due to the high phase margin constraint (Table 5.3). The one-stage op-amps have mostly a simple first stage. The three telescopic op-amps all have a simple current mirror as one of the load parts (Fig. 5.4a). The second stages in the symmetrical op-amps are mostly simple. The telescopic op-amp in Fig. 5.4a is a valid topology of *Specs 3*.

For *Specs 4*, 468 of the 936 fully-differential op-amp topologies were created in total, as 39 one-stage op-amps fulfilled the specifications in $F_{B,start}$. 34 topologies fulfilled all specifications. These are mainly folded-cascode one-stage and two-stage op-amps (Table 5.3). Also, one telescopic one-stage op-amp fulfilled the specifications. An example of a fully-differential op-amp topology fulfilling *Specs 4* is shown in Fig. 5.4b. It is a folded-cascode one-stage op-amp with a PMOS differential stage.

The number of topologies fulfilling *Specs 5* is the smallest in this scenario: Similar to *Specs 4*, 37 topologies fulfilled $F_{B,start}$ leading to 444 created op-amps. Only two folded-cascode one-stage topologies fulfilled all specifications (Table 5.3). One is the topology shown in Fig. 5.4b. The other one is an NMOS version of it. The NMOS version has a cascode stage bias in the first stage instead of a simple one.

As currently only one-stage complementary op-amps are supported by FUBOCO, all 36 topologies are created for every run specified for complementary op-amps. 10 topologies fulfill all specifications in *Specs 6*. They only vary in their type of loads and stage biases as they all have a folded-cascode first stage. The loads are mainly loads with one load part being a current mirror and the other load part containing two current biases (Fig. 5.4d). Various types of current mirrors as load fulfill the specifications. All types of stage biases appear in the topologies.

Six complementary op-amp topologies fulfill *Specs 7*. They do not differ much from the op-amps fulfilling *Specs 6*. The simple stage bias in the first stage dominates the set. The topology in Fig. 5.4d also fulfills *Specs 7*.

Please note that topologies might overfill specifications with quite high margin, as only one specification bound is given per performance feature. This can be omitted by using an upper and lower bound for each performance feature, e.g., a maximum and a minimum open-loop

Table 5.4.: Runtime comparison of the sequential algorithm (Sec. 5.4.2) and the parallel algorithm (Sec. 5.5.2) on different CPUs; (m): max. # of threads; [n] runtime in percent compared to sequential algorithm

Specs #		1	2	3	4	5	6	7
Sequential algorithm (Sec. 5.4) (1)		21 h	16 h	14.5 h	8 h	8 h	35 min	30 min
Parallel algorithm (Sec. 5.5.2)	Intel® Core™ i5-7500 CPU@3.4GHz (4)	7 h [33%]	5.5 h [34%]	5.5 h [38%]	3 h [38%]	3.5 h [44%]	15 min [43%]	14 min [47%]
	2 x Intel® Xeon® CPU E5-2695 @2.4GHz (2 x 8)	4 h [19%]	3.2 h [20%]	3.2 h [22%]	1.6 h [20%]	1.6 h [20%]	11 min [31%]	9 min [30%]
	2 x Intel® Xeon® Gold 6126 CPU @2.6GHz (2 x 24)	3.5 h [17%]	3 h [19%]	3 h [21%]	1.2 h [15%]	1.2 h [15%]	7.5 min [21%]	6 min [20%]

gain requirement. This would reduce the resulting topologies only to topologies performing very close to the specifications. A smaller number of topologies would fulfill specifications with low demands, e.g., *Specs 1*.

5.6.2. Synthesis Runtime

The search duration of the algorithm varies depending on the specification (Table 5.4), as it highly depends on the number of created topologies (Table 5.2). The sizing of an op-amp with its optimization time of one min per circuit is the biggest time constraint of the synthesis tool. In contrast, the creation of a topology is in the range of ms.

General Runtime Behavior

For a single-output op-amp specification set, many topologies can be synthesized and sized (2940). For such specifications, the runtime of the synthesis tool is quite long (14 h - 22 h for the sequential algorithm). For a complementary op-amp with only 36 topologies supported, the runtime is much smaller ($\sim$ 30 min for the sequential algorithm). The requirements of the specifications have a great influence on the runtime. If they are more strict, the runtime decreases for two reasons:

- Strict specifications lessen the number of two-stage op-amps which are created and sized. Many one-stage op-amps do not fulfill the specifications in $F_{B,start}$. Thus, two-stage op-amps based on them are not created.
- The constraint-programming solver does not optimize circuits if early results show that the circuit does not fulfill the specifications.

For *Specs 1*, 144 of 210 one-stage op-amps fulfilled the specifications in $F_{B,start}$ (5.2). Thus, 1728 op-amps in total were created by the FUBOCO synthesis process. Around 21 h were needed to evaluate all circuits on a common computer running the sequential algorithm. For *Specs 2*, only 96 one-stage op-amps fulfilled $F_{B,start}$. The total number of created op-amp topologies was reduced to 1152. This also reduced the runtime of the synthesis tool to around 16 h. For *Specs 3*, the number of one-stage op-amps fulfilling $F_{B,start}$ was even smaller, and thus also the evaluation time. For fully-differential op-amps, the runtime for both specification sets is equal. The total number of one-stage op-amps is smaller compared to single-output op-amps. Hence, the number of one-stage op-amps fulfilling F_{start} does not vary as much. For the complementary op-amp, only one-stage op-amps are supported. Thus, always 36 op-amps are created and evaluated. For more challenging specs (*Specs 7*),

the runtime still decreases, as the number of topologies increases for which the sizings are not further optimized as the topologies will not fulfill the specifications.

Comparison of the Sequential Algorithm to the Parallel Algorithm

Table 5.4 shows the runtime of the sequential algorithm (Sec. 5.4.2) as well as the runtime of the parallel algorithm in Sec. 5.5.2. The parallel synthesis algorithm was tested on three different CPUs. The Intel®Core™ i5-7500 CPU@3.4GHz is a common processor for desktop computers able to run 4 threads in parallel. The Intel® Xeon® CPU E5-2695@2.4GHz and Intel® Xeon® Gold 6126 CPU@2.6GHz are two server CPUs supporting 8, respectively 24, threads in parallel. Even on a small desktop CPU, the runtime of the parallel algorithm is reduced to 40% of the sequential runtime, needing less than 8 h to find all possible topologies for general specifications. On more advanced hardware, only 20% of the sequential runtime is needed, reducing the runtime for general specifications to half a work day.

Regarding *Specs* 2, 3, the runtime of the sequential algorithm is 1.5 h faster for *Specs* 3, as fewer two-stage op-amps are created for *Specs* 3. This runtime reduction cannot be seen for the parallelized algorithms. Both specification sets have identical runtimes. The reason for this lies in the missing parallelization of the sizing and evaluation of two-stage op-amps. For *Specs* 3, the additional one-stage op-amps not fulfilling $F_{B,start}$ are unevenly distributed among threads, several of the parallel running threads have more one-stage op-amps for which two-stage op-amps are created.

The synthesis algorithm (Fig. 5.6b) creates a fixed number of op-amp topologies before the sizing and evaluation step. For CPUs supporting a high number of threads T_H, this might lead to unused threads, as the number of created op-amp topologies n is smaller than T_H. This is the case for the Intel® Xeon® Gold CPU. In many runs, not all 48 threads are used, which leads to similar runtimes as for the Intel® Xeon® E5 processor. However, creating as many op-amp topologies as number of threads T_H might exceed the memory. It remains future work to enhance the algorithm such that the number of op-amp topologies created at the same time corresponds to the amount of threads being able to run in parallel on the CPU and the assigned RAM storage to the program.

5.6.3. Sizing and Evaluation Results[7]

A selection of the outputted topologies fulfilling all specifications were simulated to analyze how the circuits sized with analytical equations agree with simulation results (Table 5.5). A BSIM3v3 transistor model was used for simulations. Table 5.5 shows the average deviations between analytically calculated and simulated values. The deviations agree well with the expectations of a designer, who expects a deviation below 30%. The largest deviation occurs for the unity-gain bandwidth. Its value is overestimated in the sizing process as it depends linearly on the transconductance of the first stage in the analytical equations (Chapter 4).

[7]A similar version of this section was published in [62], reprinted with permission from "Inga Abel, Helmut Graeb, FUBOCO: Structure Synthesis of Basic Op-Amps by FUnctional BlOck COmposition, ACM Trans. Des. Autom. Electron. Syst." © 2022 ACM.

Table 5.5.: Comparison to simulation results; Deviation in percent; (i/j), i: # of topologies fulfilling the specified value, j: total # of topologies tested

Specs #	1	2	3	4	5	6	7
Quiescent power	8% (25/30)	7% (28/28)	6% (21/23)	4% (10/10)	4% (0/2)	20% (9/10)	6% (6/6)
Phase margin	24% (25/30)	11% (28/28)	6% (16/23)	3% (10/10)	3% (2/2)	16% (10/10)	15% (5/6)
CMRR	20% (30/30)	27% (26/28)	21% (22/23)	19% (10/10)	16% (2/2)	18% (10/10)	11% (6/6)
CMIR	(30/30)	(27/28)	(23/23)	(10/10)	(2/2)	-	-
Open-loop gain	16% (19/30)	25% (21/28)	12% (22/23)	15% (9/10)	12% (2/2)	25% (9/10)	11% (6/6)
Unity-gain bandwidth	29% (21/30)	25% (22/28)	19% (19/23)	31% (5/10)	19% (2/2)	32% (9/10)	32% (6/6)
Slew rate	15% (25/30)	24% (15/28)	16% (17/23)	20% (10/10)	26% (2/2)	22% (6/10)	36% (3/6)
Output voltage swing	(24/30)	(23/28)	(19/23)	(10/10)	(2/2)	(7/10)	(4/6)
All *Specs*	(6/30)	(8/28)	(6/23)	(5/10)	(0/2)	(2/10)	(2/6)

The absolute number of circuits fulfilling a specification respective all specifications is also given. For many specification sets, already a number of topologies fulfill all specifications. Other circuits need subsequent optimization [74] to fulfill all specifications.

6. Outlook: Multi-Stage Op-Amps

Multi-stage op-amps play an important role in modern short channel technologies. However, their design is still a research issue. Fig. 6.1 shows an example of a multi-stage op-amp. In the following sections, we discuss on this example how modern multi-stage topologies are supported by the three methods described in this thesis. Sec. 6.1 discusses how the functional block decomposition method in Chapter 3 has to be extended to support multi-stage op-amp topologies. Sec. 6.2 discusses the adjustments which have to be performed on the sizing method in Chapter 4. Extensions on the synthesis method (Chapter 5) to support also multi-stage op-amps are discussed in Sec. 6.3. Also, the limitations of the three methods with regard to multi-stage op-amps topologies are discussed.

6.1. Extension of the Functional Block Decomposition Method

The complete structural analysis of the op-amp in Fig. 6.1 is already supported by the functional block decomposition method in Chapter 3 (Sec. 3.3). The op-amp has two capacitor structures for frequency compensation. More advanced frequency compensation methods, e.g., [78, 79], can also be supported by the method, as they have fixed structures. Other important compensation structures in multi-stage op-amps are feedback stages, e.g., [76, 77, 99]. They have a similar structure as inverting stages with a voltage bias as stage bias (3.36) and thus, can be identified with a similar structural description.

6.2. Extension of the Hierarchical Performance Equation Library

To support the above discussed structures in [76–79, 99] by the sizing method in Chapter 4, their behavior descriptions must be added to the hierarchical performance equation library (HPEL, Sec. 4.2). For the compensation structures in [78, 79], the behavior descriptions are given in the corresponding papers. To support multi-stage op-amps in general in the sizing method, the HPEL must be extended to support additional inverting stages. For most

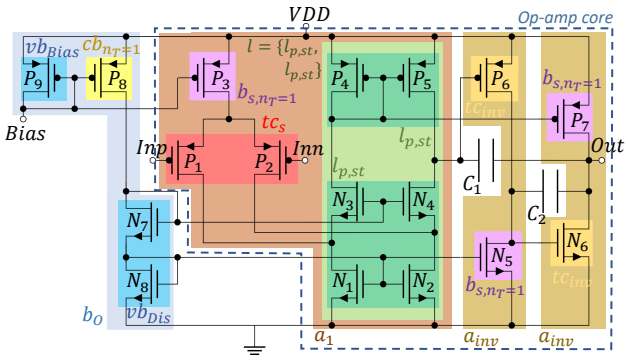


Figure 6.1.: Three-stage op-amp, colored background: functional blocks of HL 2 - 4

Table 6.1.: Results of the sizing method in Chapter 4 for the three-stage op-amp (Fig. 6.1)

Constraints	Spec.	M	S
Bias current (μA)	10	10	10
Load capacity (pF)	20	20	20
Supply voltage (V)	5	5	5
Gate-area (10^3 μm^2)	$\leq$15	8.3	-
Quiescent power (mW)	$\leq$20	15	17
Phase margin (°)	$\geq$60	-	50
CMRR (dB)	$\geq$70	-	180
CMIR	2-3	0-4.1	0.4-4
Open-loop gain (dB)	$\geq$60	88	73
Unity-gain bandwidth (MHz)	$\geq$2.5	2.8	1.9
Slew rate ($\frac{V}{\mu s}$)	$\geq$2.5	2.6	8.4
Output voltage swing (V)	1-4	0.3-4.3	0.2-4.2

(a) Performance values; M: sizing method; S: simulation

Variable	Value (μm/fF)
$w_{P_{1,2}}; l_{P_{1,2}}$	6;3
$w_{P_3}; l_{P_3}$	13;6
$w_{P_{4,5}}; l_{P_{4,5}}$	15;1
$w_{P_6}; l_{P_6}$	157;9
$w_{P_7}; l_{P_7}$	256;1
$w_{P_8}; l_{P_8}$	203;6
$w_{P_9}; l_{P_9}$	104;6
$w_{N_{1,2}}; l_{N_{1,2}}$	319;3
$w_{N_{3,4}}; l_{N_{3,4}}$	222;3
$w_{N_5}; l_{N_5}$	189;3
$w_{N_6}; l_{N_6}$	570;1
$w_{N_7}; l_{N_7}$	37;3
$w_{N_8}; l_{N_8}$	41;3
c_1	5
c_2	100

(b) Device sizes

performance features, as open-loop gain (4.44) or slew rate (4.45), this means only marginal changes in the equation set-up. This is different for the phase margin. The generation of good models for the phase margin of multi-stage op-amps is still a research issue in analog design [100]. No method is currently suitable to be integrated into the HPEL. To nevertheless generate sizing results for multi-stage op-amps, common stability constraints as given in [76, 77] can be used. Table 6.1 shows sizing results for the three-stage op-amp in Fig. 6.1 obtained with such constraints. The compensation capacitors were manually adopted to improve the stability of the op-amp. As the sizing tool provides good values for all specifications not affected by the changed capacitor values (Table 6.1a), the sizing method can be used to size multi-stage op-amps as presented in [76, 77]. The sizing results will improve when adequate models for the phase margin of multi-stage op-amps are found. Please note that changes on the constraint programming solver are not needed to support multi-stage op-amps in the sizing method as the branching heuristic is dynamic (Sec. 4.5).

6.3. Extension of the Synthesis Method

To add the generation of multi-stage op-amps to the synthesis method in Chapter 5, the new aspects of the topologies must be added to the method. In general, this means adding an additional third-stage to two-stage op-amp topologies. The three-stage op-amp in Fig. 6.1 and similar topologies can, e.g., be synthesized by following input to Alg. 5:

$S_1 : OP_{so,2,\backslash b_O \backslash cap}$; $S_2 : A_{inv}$; $S_3 : cap$; $S_4 : cap$; $R_c :$ $s_1.out \leftrightarrow s_2.in_{tc,1}, s_1.inner_{1,l_{p,st,a_1}} \leftrightarrow s_2.in_{b_{s,1}}, s_1.out_{a_1} \leftrightarrow s_3.plus, s_1.out_{a_2} \leftrightarrow s_4.plus, s_2.out \leftrightarrow s_3.minus,$ $s_2.out \leftrightarrow s_4.minus$;
$R_f :$ $s_1.\Phi_{tc,a_1} \neq s_2.\Phi_{tc}$,$s_1.\Phi_{tc,a_2} \neq s_2.\Phi_{tc}$;

The inputted functional blocks are a set of single-output two-stage op-amp topologies without their bias circuit b_O and compensation capacitor cap, $OP_{so,2,\backslash b_O \backslash cap} = \{A_1, A_2\}$, a set of inverting stages A_{inv}, which form the third stages of the op-amps, and two sets of capacitors cap. The connection rules R_c contain that the inverting stage of the newly created topology must be connected as third stage to the output of the two-stage op-amp. The stage bias of the third stage is biased by one of the load parts of the first stage l_{p,st,a_1}. The capacitors are connected between the outputs of first and third stage, and the outputs of second and third

stage. The topologies are only valid if the transconductor of the third stage has different doping as the transconductor of first and second stage. The bias circuits of the op-amp topologies are created by Alg. 6.

Other topology variants as, e.g., provided in [76, 77] can be similarly included. The topology evaluation must be performed by an enhanced version of the sizing method described in Chapter 4 with adjustments similarly to the extensions described in Sec. 6.2. Limitations arise again from the missing adequate model of the phase margin. Hence, including multi-stage op-amps into the synthesis method described in this thesis remains future works until good and simple models describing the phase margin behavior are found.

7. Conclusion

Analog circuits are a significant cost driver for modern integrated circuits. The design process of analog circuits is in many parts still manual. This is time consuming and leads to an error-prone process. The state of the art in the automation of the analog design process is software that guides the manual design process through simulation. Optimization methods which use numerical methods to, e.g., size an analog circuit automatically as [74] are not widely used. The numerical approaches lack the physical orientation and the traceability of the manual design process. This thesis has presented an automation approach to analog circuit design that sticks closely to the manual design process but automates two important steps of the process: topology selection and sizing. The method emulates the analog design process. Knowledge from state-of-the-art analog design books is formalized in a computer comprehensible way. The method focuses on operational amplifiers. The thesis contains the following contributions compared to the state of the art:

- a functional block decomposition method which proposes a new formalization of functional blocks in op-amps. A hierarchical functional block description for op-amps was developed which builds up functional blocks hierarchically based on lower level subblocks. On the lowest hierarchy level are single devices followed by simple transistor pairs as voltage or current bias. Higher hierarchy levels contain transconductors, loads and stage biases as well as whole amplification stages and op-amp topologies. Each functional block is given a comprehensible structural and functional description. These descriptions are used in algorithms to automatically identify all functional blocks in given op-amp topologies.
- a hierarchical performance equations library (HPEL) that contains for each functional block of an op-amp a behavioral description based on analytical equation. The HPEL allows a topology independent set-up of the equation-based description of the manual op-amp design process. Using newly developed algorithms, the generic equations part of the HPEL are customized for a given topology. The performance model of the circuit generated by the HPEL describes the ac-behavior, dc-behavior, and transient behavior of the circuit with the same equations as in the manual design process.
- an automatic equation-based sizing process that follows closely the manual design process. The circuit model instantiated with the HPEL is suitable for circuit sizing. Constraint programming is used to solve it. To improve the reliability of the constraint programming solver, we developed a problem specific branching heuristic.
- a structural synthesis method based on functional blocks. The hierarchical structure description of functional blocks in the functional block decomposition method is used for the structural synthesis of op-amp topologies. Based on a hierarchical composition graph, topologies are synthesized based on their containing functional blocks. A generic algorithm has been presented to generate the transistor implementations of a functional block based on the implementation of its subblocks. Another algorithm has been presented to create a customized bias for an op-amp topology. The created topologies are evaluated for a given specification set using the automatic equation-based sizing

method which was enhanced for structural op-amp synthesis. Thousands of op-amp topologies are supported.

Experimental results showed the effectiveness of each method. Each method supports single-output, fully-differential and complementary op-amps. The topologies can have one or two amplification stages. Also symmetrical op-amps are supported. The functional block decomposition method additionally supports three-stage op-amps. The runtime of the functional block decomposition method is insignificant and lies in the area of ms. Also the runtime of the algorithms to instantiate an equation-based circuit model for given topologies is in the area of ms. The runtime of the constraint programming solver to size a circuit varies between a few seconds and one minute depending on the structural complexity of the topology to be sized. The runtime of the structural synthesis algorithm highly depends on the given specification set and the used hardware as it uses multi-threading to parallelize part of the algorithm. A synthesis run featuring several thousand op-amp topologies has a duration of less than four hours on average.

The hierarchical structure of the presented methods allows an easy extension to support other op-amp topologies and analog circuit classes. New functional blocks can be added to the functional block decomposition method by developing a structural description for them. Already available structural descriptions for subblocks can be reused. To support the sizing of a new functional block, the hierarchical performance equation library (HPEL) must be extended by adding the analytical performance model of the new functional block to the HPEL. Lower level equations of the HPEL can be reused. Many of the basic equations are already integrated in the HPEL. As many structural and behavioral descriptions of functional blocks are reusable, the time needed to add a new functional block to the functional block decomposition method or the hierarchical performance equation library is comparably fast and needs half a work day on average. To add new topologies to the structural synthesis algorithm, only the new aspects of the topology must be added to the method with the corresponding synthesis rules. Thus, whole sets of topologies can be included in the method with only a small set-up effort.

A. Additional Equations of the Hierarchical Performance Equation Library

This appendix presents additional equations of the hierarchical performance equation library. These are equations to calculate the capacitance of a transistor at a specified pin (Sec. A.1) and equations of important non-dominant poles in op-amps (Sec. A.2) which should be considered during phase margin calculation. Also, equations of important zeros are stated (Sec. A.3) which should be considered during phase margin calculation for certain op-amp topologies.

A.1. Capacitance at a Transistor Pin

The capacitance C_{p_j} arising at a pin of a transistor is calculated by:

$$C_{p_j} = \begin{cases} C_{cap}, & p_j \in P_{cap} \\ C_{GB} + C_{GS} + C_{GD}, & p_j.type = gate \\ C_{GD} + C_{DB}, & p_j.type = drain \\ C_{GS} + C_{SB}, & p_j.type = source \end{cases} \tag{A.1}$$

C_{cap} is the capacitance of a capacitor *cap* connected with one of its pins to the net. For the calculation of the gate-source capacitance C_{GS}, gate-drain capacitance C_{GD}, gate-bulk capacitance C_{GB}, drain-bulk capacitance C_{DB}, and source-bulk capacitance C_{SB}, textbook equations as, e.g., in [80], are used.

Please note that if the drain and the gate of a transistor t_k are connected to n_j, which is the case if $t_k.type = dt$, the gate-drain capacitance C_{GD} is only added once to C_{n_i}.

A.2. Important Non-Dominant Poles in Op-Amps

All poles stated in this section have a noticeable influence on the phase margin. Not named poles have either a high frequency, so that their effect on the phase margin is insignificant, or are close to other poles or zeros, so that only one of the poles has an effect on the phase margin.

Non-dominant pole of the non-inverting stage $f_{ndp,ninv}$: Every non-inverting stage has a non-dominant pole relevant for the phase margin. The calculation of the pole depends on the inner structure of the load part $l_{p,j}$ connected to the non-inverting transconductor tc_{ninv}.

$$f_{ndp,ninv} = \begin{cases} \frac{gml_{p,j,out_2}}{2\pi C_{n_{tc,ninv,out_2}}}, & l_{p,j} = \{cm_j\} \\ \frac{gml_{p,j,out_i}}{2\pi C_{n_{tc,ninv,out_i}}}, & l_{p,j} = (\{cb_{j,1}, cb_{j,2}\} \\ & \quad \vee \{vb_{j,1}, vb_{j,2}\}) \end{cases} \tag{A.2}$$

If a current mirror cm_j as load part $l_{p,j}$ is connected to tc_{ninv}, for $gm_{l_{p,j,out_2}}$ the transistor of $l_{p,j}$ is chosen, which is connected with its gate to tc_{ninv} by the net $n_{tc,ninv,out_2}$. The net $n_{tc,ninv,out_2}$ does not represent the output net of the op-amp or is connected to a further stage of the op-amp.

If the load part $l_{p,j}$ consists of voltage or current biases, it does not matter which output net of the transconductor is chosen to calculate $C_{n_{tc,ninv,out_i}}$, as the nets are symmetrical with equal capacitances. For $gm_{l_{p,j,out_i}}$, a transistor of $l_{p,j}$ is chosen which is connected with its drain to an output net $n_{out,j}$ of the first stage. As the transistor can be part of a gate-connected couple gcc, $n_{out,j}$ and $n_{tc,ninv,out_i}$ might be different. This is, for example, the case in the folded-cascode op-amp (Fig. 4.2b). The non-dominant pole of its first stage $f_{ndp,1}$ is calculated by:

$$f_{ndp,1} = \frac{gm_{P_2}}{2\pi C_{n_3}} \tag{A.3}$$

If the first stage is complementary, two non-dominant first stage poles exist.

Non-dominant pole of inverting stages $f_{ndp,inv}$: For inverting stages, three non-dominant pole types exist. The first type is provoked if a compensation capacitor is connected between the inverting stage and a previous stage. It is calculated by (4.27) stated in Sec. 4.2.6:

$$f_{ndp,inv,C_c} = \frac{gin_{tc_{inv}}}{2\pi(C_{n_{out}} + \frac{C_{n_{tc,inv,in,g}} \cdot C_{n_{out}}}{C_C} + C_{n_{tc,inv,in,g}})} \tag{A.4}$$

$gin_{tc_{inv}}$ is the transconductance of the stage. $C_{n_{tc,inv,in,g}}$ is the capacitance of the gate net carrying the input signal of the stage. $C_{n_{out}}$ is the capacitance of the output net of the stage and C_C the capacitance value of the compensation capacitor. Thus, in the symmetrical op-amp with high PSRR (Fig. 4.2c), the non-dominant pole of the third stage $f_{ndp_{a_3}}$ equals to:

$$f_{ndp,a_3} = \frac{gm_{N_6}}{2\pi(C_{n_{Out}} + \frac{C_{n_6} C_{n_{Out}}}{C_C} + C_{n_6})} \tag{A.5}$$

Non-dominant pole of a second stage with voltage bias as stage bias $f_{ndp,sym}$*:* In symmetrical op-amps, the second stage with voltage bias as stage bias evokes a non-dominant pole. It depends on the structure of this second stage transconductor $tc_{inv,vb}$:

$$f_{ndp,sym} = \begin{cases} \frac{gm_{tc,inv,vb,out}}{2\pi C_{n_{tc,inv,vb,out,s}}}, & tc_{inv,vb} = \{ts_k\} \wedge |ts_k| = 2 \\ \frac{gm_{b,s,inv,vb,out}}{2\pi C_{n_{a,inv,vb,out}}}, & tc_{inv,vb} = \{ts_k\} \wedge |ts_k| = 1 \end{cases} \tag{A.6}$$

If $tc_{inv,vb}$ consists of two transistors, $f_{ndp,sym}$ is calculated by the transistor of $tc_{inv,vb}$ connected to the output of the second stage. The capacitance $C_{n_{tc,inv,vb,out,s}}$ is the capacitance of the source net of this transistor. If $tc_{inv,vb}$ consists of one transistor, $f_{ndp,sym}$ is calculated by the transistor of the stage bias $b_{s,inv,vb}$ of the inverting stage $a_{inv,vb}$ being a voltage bias and connected with its gate to the output of the stage. The capacitance $C_{n_{a,inv,vb,out}}$ is the capacitance of the output net of the stage.

In the symmetrical op-amp (Fig. 4.2c), $tc_{inv,vb}$ consists of two transistors P_3, P_5. Thus, $f_{ndp,sym}$ is equal to:

$$f_{ndp,sym} = \frac{gm_{P_5}}{2\pi C_{n_4}} \tag{A.7}$$

Non-dominant pole of the cascode transconductor of an inverting stage $f_{ndp,inv,2}$: If an inverting stage has a cascode transconductor $tc_{inv} = \{ts_k\} \wedge |ts_k| = 2$ and is not a second stage in a symmetrical op-amp, an additional pole must be calculated. The calculation is similar to the calculation for a cascode transconductor in (A.6):

$$f_{ndp,inv,2} = \frac{gm_{tc,inv,out}}{2\pi C_{n_{tc,inv,out,s}}} \tag{A.8}$$

$gm_{tc_{inv,out}}$ is the transconductance of the transistor of tc_{inv} connected with its drain to the output of the inverting stage a_{inv}. $C_{n_{tc,inv,out,s}}$ is the capacitance of the source net of this transistor.

A.3. Important Zeros in Op-Amps

Two types of zeros have a noticeable influence on the phase margin.

Zero of non-inverting stages $f_{z,ninv}$: A non-dominant pole acting as mirror pole occurs in non-inverting stages, if the load part directly connected to the non-inverting transconductor is a current mirror. The corresponding zero is then:

$$f_{z,ninv} = 2 \cdot f_{ndp,ninv} \tag{A.9}$$

The Miller op-amp in Fig. 4.4 has such a zero.

Zero in symmetrical op-amps $f_{z,sym}$: If each of the two second-stage transconductors in a symmetrical op-amp consists of only one transistor, the non-dominant pole calculated for the second stages influences only half of the signal as the non-dominant pole $f_{ndp_{sym}}$ only occurs in the second stage with voltage bias as stage bias. Thus, the corresponding zero is:

$$f_{z,sym} = 2 \cdot f_{ndp,sym} \tag{A.10}$$

In the symmetrical op-amp in Fig. 4.2c, the transconductors of the second stages consist of two transistors. Thus, in both of the transconductors, half of the signal is influenced by the non-dominant pole in these transconductors. No zero is needed.

B. Implementation Rules for Additional Functional Blocks[1]

In the following sections, functional blocks and their composition rules are presented which are needed to support symmetrical single-output op-amps (Sec. B.1), fully-differential op-amps (Sec. B.2) and complementary op-amps (Sec. B.3).

.

B.1. Symmetrical Op-Amps ($op_{SO,sym}$)

Fig. B.1 shows the functional blocks and their composition rules for symmetrical op-amps.

Voltage bias load part implementations ($l_{p,vb}$, Fig. B.1a, HL 3) are synthesized based on two identical voltage bias implementations (vb, Fig. 5.4c $P_1 - P_4$) ($S_1 : VB_\Phi$; $S_2 : VB_\Phi$). The two voltage biases are connected at their sources ($R_c : s_1 = s_2, s_1.source \leftrightarrow s_2.source$). $\mathcal{R}_f : s_1.out_1 \leftrightarrow s_1.in$ omits mixed pairs (Fig. 5.1) as voltage bias.

Symmetrical non-inverting stages (a_{sym}, Fig. B.1b, HL 4) feature one load part consisting of voltage biases l_{p,vb,Φ_2} as load. The outputs of the simple transconductor tc_s are connected to the input pins of the load.

$S_1 : TC_{s,\Phi_1}$; $S_2 : B_{s,\Phi_1}$; $S_3 : L = \{L_{p,vb,\Phi_2}\}$;
$R_c : s_1.source \leftrightarrow s_2.out$, $s_1.out_1 \leftrightarrow s_3.in_1$, $s_1.out_2 \leftrightarrow s_3.in_2$;

Symmetrical op-amps ($op_{SO,sym}$) (Fig. B.1c, HL 5) are synthesized by adding an inverting stage a_{inv} and an inverting stage with a voltage bias as stage bias $a_{inv,vb}$ to the outputs of a symmetrical non-inverting stage a_{sym}. The transconductors tc_{inv}, $tc_{inv,vb}$ of the inverting stages $a_{inv}, a_{inv,vb}$ must have the same doping as the load part of the non-inverting stages. The number of transistors must be equal in $tc_{inv}, tc_{inv,vb}$. The transistor sum in $tc_{inv}, tc_{inv,vb}$ must be greater than or equal to the number of transistors in the load part $l_{p,vb}$ of a_{sym}. The two stage biases $b_{s,inv}, b_{s,inv,vb}$ of $a_{inv}, a_{inv,vb}$ must form a current mirror cm.

$S_1 : A_{sym,l_{p,vb}.\Phi=\Phi_1}$; $S_2 : A_{inv,tc_{inv}.\Phi=\Phi_1}$; $S_3 : A_{inv,vb,tc_{inv}.\Phi=\Phi_1}$; $R_c : n_{T,tc_{inv}} = n_{T,tc_{inv,vb}}$,
$n_{T,l_{p,vb}} \leq (n_{T,tc_{inv}} + n_{T,tc_{inv,vb}})$, $s_1.out_{ts,1,1} \leftrightarrow s_2.in_{tc,1}$, $s_1.out_{ts,2,1} \leftrightarrow s_3.in_{tc,1}$,
$s_2.in_{b_s,1} \leftrightarrow s_3.out_{b_s,1}$, $\forall_{n_{T,l_{p,vb}}=2,n_{T,tc_{inv}}=2}$ $s_2.in_{tc,2} \leftrightarrow s_3.in_{tc,2}$,
$\forall_{n_{T,l_{p,vb}}=4,n_{T,tc_{inv}}=2}[(s_1.out_{ts,1,2} \leftrightarrow s_2.in_{tc,2}) \wedge (s_1.out_{ts,2,2} \leftrightarrow s_3.in_{tc,2}))], n_{T,b_{s,inv}} \geq n_{T,b_{s,inv,vb}}$,
$\forall_{n_{T,b_{s,inv,vb}}=2}$ $s_2.in_{b_s,2} \leftrightarrow s_3.out_{b_s,2}$; $\mathcal{R}_f : \mathcal{R}_{f,b_{s,inv},b_{s,inv,vb}} = \mathcal{R}_{f,cm}$

The bias b_O is generated with Alg. 6.

[1]The appendix was similarly published in [62], reprinted with permission from "Inga Abel, Helmut Graeb, FUBOCO: Structure Synthesis of Basic Op-Amps by FUnctional BlOck COmposition, ACM Trans. Des. Autom. Electron. Syst." © 2022 ACM.

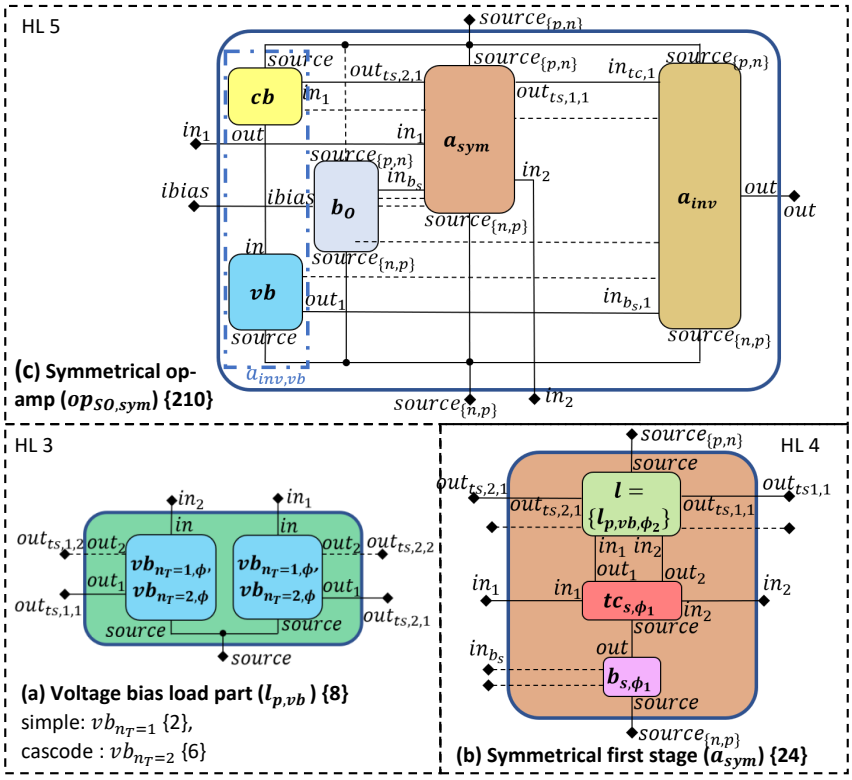


Figure B.1.: Composition rules for additional functional blocks needed to synthesize symmetrical op-amps, $\{n\}$ denotes the respective number of synthesizable structural implementations

B.2. Fully-Differential Op-Amps (op_{FD})

Fig. B.2 shows additional functional blocks and their composition rules for fully-differential op-amps.

Common-mode feedback transconductor implementations (tc_{CMFB}, Fig. B.2a, HL 3) are generated based on two differential pairs with equal doping Φ ($S_1 : DP_\Phi$; $S_2 : DP_\Phi$, e.g, Fig. 5.4b, $P_8 - P_{11}$). One input of both differential pairs is connected. Also, the outputs are connected ($R_c : s_1.in_2 \leftrightarrow s_2.in_1,\ s_1.out_1 \leftrightarrow s_2.out_2, s_1.out_2 \leftrightarrow s_2.out_1$).

Common-mode feedback stages (a_{CMFB}, Fig. B.2b, HL 4) are created based on a common-mode feedback transconductor tc_{CMFB}. The load consists of one load part with two simple voltage biases $l_{p,vb,n_T=2}$. Two identical stage biases are connected with their outputs to the respective source pins of the transconductor.

$S_1 : TC_{CMFB,\Phi_1};\ S_2 :\ L = \{L_{p,vb,n_T=2,\Phi_2}\};\ S_3 : B_{s,\Phi_1};\ S_4 : B_{s,\Phi_1};\ R_c :\ s_1.out_1 \leftrightarrow s_2.in_1,$
$s_1.out_2 \leftrightarrow s_2.in_2;\ s_1.source_1 \leftrightarrow s_3.out,\ s_1.source_2 \leftrightarrow s_4.out,\ s_3 = s_4,$

The algorithm supports one- and two-stage fully-differential op-amps. *Fully-differential one-stage op-amp* implementations ($op_{FD,1}$, Fig. B.2c, HL 5) are generated based on a first stage and a feedback stage as input to Alg. 5. The transconductors of the first stage and the feedback stage have the same doping Φ_1. The outputs of the first stage are connected to the inputs of the common-mode feedback stage. The output of the feedback stage is connected

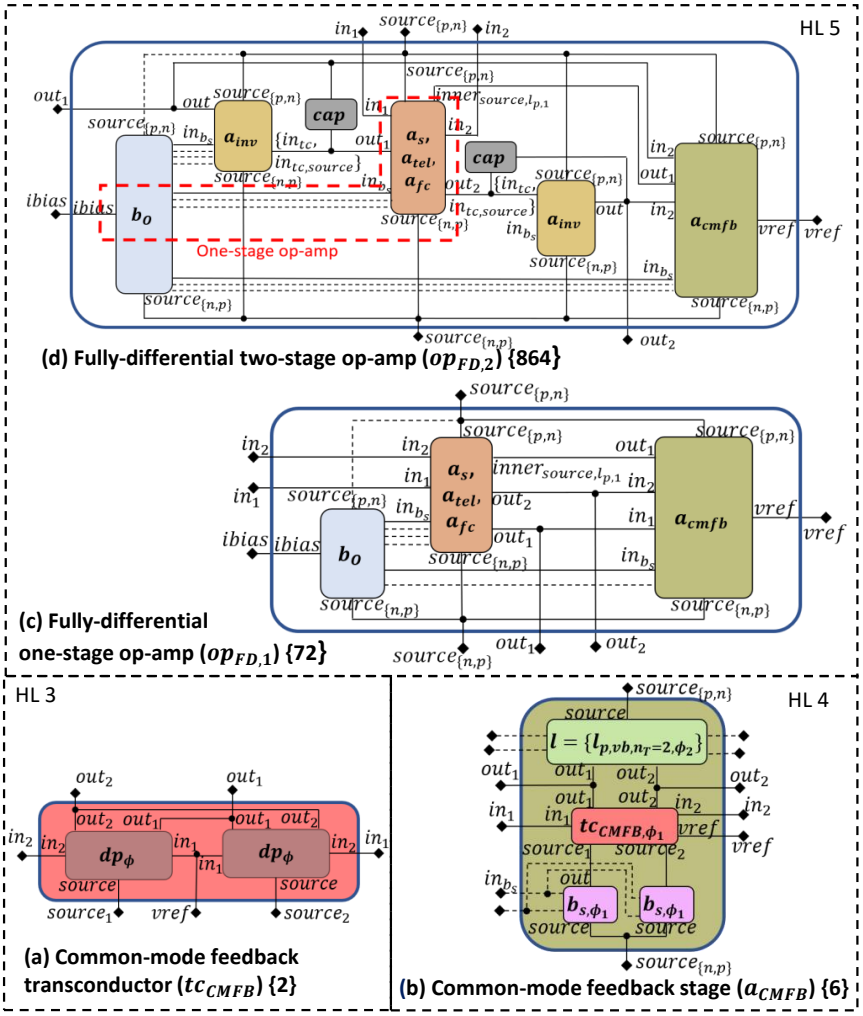


Figure B.2.: Composition rules for additional functional blocks needed to synthesize fully-differential op-amps, $\{n\}$ denotes the respective number of synthesizable structural implementations

to the gates of the transistors at the source of the load part of the first stage having the same doping Φ_2 as the load of the feedback stage.

$S_1: A_{s,tel,fc,tc_{ninv}.\Phi=\Phi_1}$; $S_2: A_{CMFB,tc_{ninv}.\Phi=\Phi_1}$;
$R_c: s_1.out_1 \leftrightarrow s_2.in_1$, $s_1.out_2 \leftrightarrow s_2.in_2$, $s_1.inner_{1,l_{p,\Phi_2}} \leftrightarrow s_2.out_1$;

The bias b_O is generated with Alg. 6.

Fully-differential two-stage op-amps ($op_{FD,2}$, Fig. B.2d, HL 5) are synthesized by adding an inverting stage and a capacitor to each output of a first stage. The inverting stages are symmetrical. Their outputs are input to the feedback stage. The output of the feedback stage is fed to the gates of the transistors at the source of the load part of the first stage having the same doping as the load of the feedback circuit. The transconductors of the feedback circuit and the first stage have the same doping. The input to Alg. 5 is:

$S_1: A_{s,tel,fc,tc_{ninv}.\Phi=\Phi_1}$; $S_2: cap$; $S_3: cap$; $S_4: A_{inv}$; $S_5: A_{inv}$; $S_6: A_{CMFB,tc_{ninv}.\Phi=\Phi_1}$;
$R_c: s_4 = s_5$, $s_1.out_1 \leftrightarrow s_2.plus$, $s_1.out_2 \leftrightarrow s_3.plus$, $s_1.out_1 \leftrightarrow s_4.in_{tc,1}$,$s_1.out_2 \leftrightarrow s_5.in_{tc,1}$,
$s_2.minus \leftrightarrow s_4.out$,$s_3.minus \leftrightarrow s_5.out$,$s_2.out \leftrightarrow s_6.in_1$,$s_3.out \leftrightarrow s_6.in_2$,
$s_1.inner_{1,l_{p,\Phi_2}} \leftrightarrow s_6.out_1$;

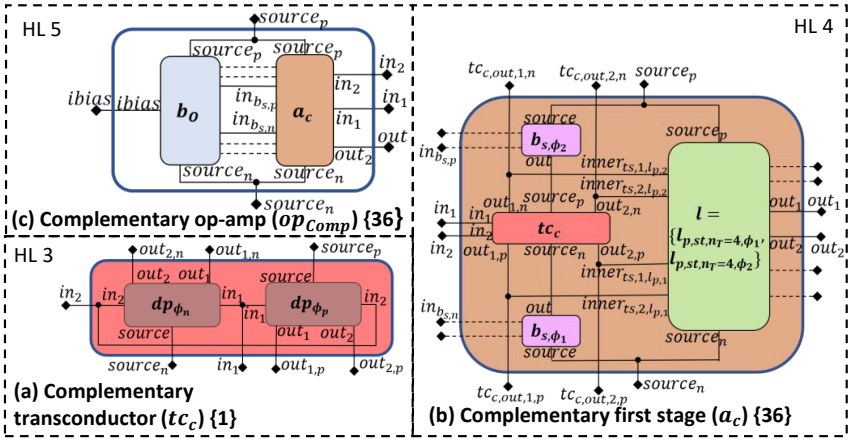


Figure B.3.: Composition rules for additional functional blocks needed to synthesize complementary op-amps, $\{n\}$ denotes the respective number of synthesizable structural implementations

The bias b_O is generated with Alg. 6.

B.3. Complementary Op-Amps (op_{comp})

Fig. B.3 shows additional functional blocks and their composition rules for complementary op-amps.

Complementary transconductors (tc_c, Fig. B.3a, HL 3) are synthesized based on two differential pairs ($S_1 : DP_{\Phi_1}, S_2 : DP_{\Phi_2}$) having different dopings Φ_1, Φ_2 (Fig. 5.4d, N_1, N_2, P_1, P_2). Both inputs of the differential pairs are connected ($R_c : s_1.in_1 \leftrightarrow s_2.in_1,\ s_1.in_2 \leftrightarrow s_2.in_2$).

Complementary first stages (a_c, Fig. B.3b, HL 4) are created based on a complementary transconductor tc_c. As tc_c has two source pins of different doping, two stage biases $b_{s,\Phi_1}, b_{s,\Phi_2}$ of different doping are connected with their outputs to the sources of the same doping. The stage biases should be symmetrical, i.e., have the same structural implementation by different doping. The load is a two-load-part load with eight transistors. The inner pins $inner_{ts_i,l_{p,j}}$ of current or voltage biases in the load are connected to the output of the transconductor having a different doping as the load part.

$S_1 : TC_c;\ S_2 : B_{s,\Phi_1};\ S_3 : B_{s,\Phi_2};\ S_4 : L_{n_{l_p}=2}\ \ R_c :\ \mathrm{sym}(s_2, s_3),\ s_1.source_{\Phi_1} \leftrightarrow s_2.out,$
$s_1.source_{\Phi_2} \leftrightarrow s_3.out,\ s_1.out_{1,\Phi_1} \leftrightarrow s_4.inner_{ts_1,l_{p,2}},\ s_1.out_{2,\Phi_1} \leftrightarrow s_4.inner_{ts_2,l_{p,2}},$
$s_1.out_{1,\Phi_2} \leftrightarrow s_4.inner_{ts_1,l_{p,1}},\ s_1.out_{2,\Phi_2} \leftrightarrow s_4.inner_{ts_2,l_{p,1}};$

Complementary op-amps (op_{Comp}) (Fig. 5.5s) are synthesized using complementary first stages (S_1: A_c). The topology specific op-amp bias is created with Alg. 6.

List of Figures

List of Tables

Bibliography

[1] (2015) World Semiconductor Trade Statistics. https://www.wsts.org/. World Semiconductor Trade Statistics. [Online]. Available: https://www.wsts.org/

[2] (2011) International Technology Roadmap for Semiconductors. https://www.itrs.org/. International Technology Roadmap for Semiconductors. [Online]. Available: https://www.itrs.net/

[3] J. Horgan, "Analog," *EDA Weekly*, March 2005.

[4] H. Graeb, "ITRS 2011 Analog EDA Challenges and Approaches," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2012, pp. 1150–1155.

[5] *Cadence*, www.cadence.com, Cadence Design Systems, 2018. [Online]. Available: www.cadence.com

[6] E. Martens and G. Gielen, "Classification of analog synthesis tools based on their architecture selection mechanisms," *Integration, the VLSI Journal*, vol. 41, no. 2, pp. 238 – 252, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167926007000417

[7] M. G. R. Degrauwe, O. Nys, E. Dijkstra, J. Rijmenants, S. Bitz, B. L. A. G. Goffart, E. A. Vittoz, S. Cserveny, C. Meixenberger, G. van der Stappen, and H. J. Oguey, "IDAC: An interactive design tool for analog CMOS circuits," *IEEE Journal of Solid-State Circuits*, 1987.

[8] F. El-Turky and E. Perry, "BLADES: An artificial intelligence approach to analog circuit design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1989.

[9] H. Y. Koh, C. H. Sequin, and P. R. Gray, "OPASYN: a compiler for CMOS operational amplifiers," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1990.

[10] F. Leyn, W. Daems, G. Gielen, and W. Sansen, "Analog Circuit Sizing with Constraint Programming Modeling and Minimax Optimization," in *IEEE International Symposium on Circuits and Systems*, 1997.

[11] G. Van der Plas, G. Debyser, F. Leyn, K. Lampaert, J. Vandenbussche, G. Gielen, W. Sansen, P. Veselinovic, and D. Leenaerts, "AMGIE–A Synthesis Environment for CMOS Analog Integrated Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2001.

[12] P. C. Maulik and L. R. Carley, "Automating analog circuit design using constrained optimization techniques," in *IEEE/ACM International Conference on Computer-Aided Design*, 1991.

[13] C.-J. R. Shi and X.-D. Tan, "Canonical Symbolic Analysis of Large Analog Circuits with Determinant Decision Diagrams," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2000.

[14] W. Verhaegen and G. Gielen, "Efficient DDD-based Symbolic Analysis of Large Linear Analog Circuits," in *ACM/IEEE Design Automation Conference*, 2001.

[15] X.-X. Liu, A. A. Palma-Rodriguez, S. Rodriguez-Chavez, S. X.-D. Tan, E. Tlelo-Cuautle, and Y. Cai, "Performance bound and yield analysis for analog circuits under process variations," in *Asia and South Pacific Design Automation Conference*, 2013.

[16] W. Nye, D. C. Riley, A. Sangiovanni-Vincentelli, and A. L. Tits, "DELIGHT.SPICE: an optimization-based system for the design of integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1988.

[17] E. S. Ochotta, R. A. Rutenbar, and L. R. Carley, "Synthesis of High-Performance Analog Circuits in ASTRX/OBLX," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1996.

[18] R. Phelps, M. Krasnicki, R. A. Rutenbar, L. R. Carley, and J. R. Hellums, "Anaconda: Simulation-Based Synthesis of Analog Circuits Via Stochastic Pattern Search," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2000.

[19] B. Liu, F. V. Fernandez, and G. Gielen, "Efficient and Accurate Statistical Analog Yield Optimization and Variation-Aware Circuit Sizing Based on Computational Intelligence Techniques," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2011.

[20] M. Eick and H. E. Graeb, "MARS: Matching-Driven Analog Sizing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2012.

[21] F. Gong, H. Yu, Y. Shi, and L. He, "Variability-Aware Parametric Yield Estimation for Analog/Mixed-Signal Circuits: Concepts, Algorithms, and Challenges," *IEEE Design and Test*, 2014.

[22] G. Berkol, E. Afacan, G. Dundar, A. E. Pusane, and F. Baskaya, "A novel yield aware multi-objective analog circuit optimization tool," in *IEEE International Symposium on Circuits and Systems*, 2015.

[23] F. Passos, E. Roca, J. Sieiro, R. Fiorelli, R. Castro-Lopez, J. M. Lopez-Villegas, and F. V. Fernandez, "A Multilevel Bottom-up Optimization Methodology for the Automated Synthesis of RF Systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019.

[24] A. Canelas, R. Povoa, R. Martins, N. Lourenco, J. Guilherme, J. P. Carvalho, and N. Horta, "FUZYE: A Fuzzy C-Means Analog IC Yield Optimization using Evolutionary-based Algorithms," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019.

[25] R. Schwencker, J. Eckmueller, H. Graeb, and K. Antreich, "Automating the sizing of analog CMOS circuits by consideration of structural constraints," in *Design, Automation and Test in Europe Conference*, 1999.

[26] M. d. Hershenson, S. P. Boyd, and T. H. Lee, "Optimal design of a CMOS op-amp via geometric programming," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2001.

[27] E. Yilmaz and G. Dundar, "Analog Layout Generator for CMOS Circuits," *IEEE Transactions onoften Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 1, pp. 32–45, Jan 2009.

[28] H. Habal and H. Graeb, "Constraint-Based Layout-Driven Sizing of Analog Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 8, pp. 1089–1102, Aug 2011.

[29] R. Martins, N. Lourenço, A. Canelas, R. Póvoa, and N. Horta, "AIDA: Robust layout-aware synthesis of analog ICs including sizing and layout generation," in *2015 International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, 2015, pp. 1–4.

[30] S. Bhattacharya, N. Jangkrajarng, and C. . R. Shi, "Multilevel symmetry-constraint generation for retargeting large analog layouts," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 6, pp. 945–960, 2006.

[31] R. Castro-Lopez, O. Guerra, E. Roca, and F. V. Fernandez, "An Integrated Layout-Synthesis Approach for Analog ICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 7, pp. 1179–1189, 2008.

[32] Koh Han Young, C. H. Sequin, and P. R. Gray, "Automatic layout generation for CMOS operational amplifiers," in *[IEEE International Conference on Computer-Aided Design (ICCAD-89) Digest of Technical Papers*, 1988, pp. 548–551.

[33] P. Lin, Y. Chang, and S. Lin, "Analog Placement Based on Symmetry-Island Formulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 6, pp. 791–804, 2009.

[34] E. S. Ochotta, R. A. Rutenbar, and L. R. Carley, "Synthesis of high-performance analog circuits in ASTRX/OBLX," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 3, pp. 273–294, March 1996.

[35] J. R. Koza, F. H. Bennett, D. Andre, M. A. Keane, and F. Dunlap, "Automated synthesis of analog electrical circuits by means of genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 2, pp. 109–128, 1997.

[36] P. C. Maulik, L. R. Carley, and R. A. Rutenbar, "Integer programming based topology selection of cell-level analog circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 4, pp. 401–412, April 1995.

[37] C. Ferent and A. Doboli, "Novel circuit topology synthesis method using circuit feature mining and symbolic comparison," in *2014 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2014, pp. 1–4.

[38] A. Gerlach, J. Scheible, T. Rosahl, and F. Eitrich, "A generic topology selection method for analog circuits with embedded circuit sizing demonstrated on the OTA example," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, 2017, pp. 898–901.

[39] R. Harjani, R. A. Rutenbar, and L. Carley, "OASYS: A Framework for Analog Circuit Synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1989.

[40] T. McConaghy, P. Palmers, M. Steyaert, and G. G. E. Gielen, "Variation-Aware Structural Synthesis of Analog Circuits via Hierarchical Building Blocks and Structural Homotopy," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 9, pp. 1281–1294, 2009.

[41] D. Guilherme, J. Guilherme, and N. Horta, "Automatic topology selection and sizing of Class-D loop-filters for minimizing distortion," in *2010 XIth International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD)*, 2010, pp. 1–4.

[42] F. Jiao and A. Doboli, "A low-voltage, low-power amplifier created by reasoning-based, systematic topology synthesis," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, pp. 2648–2651.

[43] Z. Zhao and L. Zhang, "An Automated Topology Synthesis Framework for Analog Integrated Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 12, pp. 4325–4337, 2020.

[44] A. Das and R. Vemuri, "A graph grammar based approach to automated multi-objective analog circuit design," in *2009 Design, Automation Test in Europe Conference Exhibition*, 2009, pp. 700–705.

[45] M. Meissner and L. Hedrich, "FEATS: Framework for Explorative Analog Topology Synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2015.

[46] F. Fernandez, . Rodríguez-Vázquez, J. L. Huertas, and G. G. E. Gielen, *Structural Synthesis and Optimization of Analog Circuits*, 1998, pp. 211–232.

[47] W. M. C. Sansen, *Analog Design Essentials*. Springer, 2007.

[48] K. R. Laker and W. M. C. Sansen, *Design of analog integrated circuits and systems*. McGraw-Hill, 1994.

[49] P. R. Gray, R. G. Meyer, P. J. Hurst, and S. H. Lewis, *Analysis and Design of Analog Integrated Circuits*, 4th ed. USA: John Wiley & Sons, Inc., 2001.

[50] K. M. David Johns, *Analog Integrated Circuit Design*. John Wiley and Sons, 1997.

[51] D. R. H. Phillip E. Allan, *CMOS Analog Circuit Design*. Oxford University Press, 2012.

[52] T. Massier, H. Graeb, and U. Schlichtmann, "The Sizing Rules Method for CMOS and Bipolar Analog Integrated Circuit Synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2008.

[53] D. Stefanovic and M. Kayal, *Structured Analog CMOS Design*. Springer Netherlands, 2008.

[54] M. del Mar Hershenson, S. P. Boyd, and T. H. Lee, "GPCAD: a tool for CMOS op-amp synthesis," in *1998 IEEE/ACM International Conference on Computer-Aided Design*, Nov 1998.

[55] I. Abel, M. Neuner, and H. Graeb, "A Functional Block Decomposition Method for Automatic Op-Amp Design," *Integration*, vol. 85, pp. 108–120, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167926022000438

[56] I. Abel, M. Neuner, and H. Graeb, "Automatische Initialdimensionierung von analogen Operationsverstärkern," in *VDE/VDI-GMM MikroSystemTechnik Kongress*, 2019.

[57] ——, "Constraint-Programmed Initial Sizing of Analog Operational Amplifiers," in *IEEE 37th International Conference on Computer Design (ICCD)*, 2019, pp. 413–421.

[58] I. Abel, M. Neuner, and H. Graeb, "COPRICSI: COnstraint-PRogrammed Initial Circuit SIzing," *Integration*, vol. 76, pp. 148 – 158, 2021.

[59] ——, "A Hierarchical Performance Equation Library for Basic Op-Amp Design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2021.

[60] I. Abel and H. Graeb, "Automatic Initial Sizing of Operational Amplifiers with Support of Weak, Moderate and Strong Inversion," in *17th ITG/GMM-Symposium Design of Analog Circuits with EDA Methods (ANALOG)*, 2020, pp. 1–4.

[61] ——, "Structural Synthesis of Operational Amplifiers Based on Functional Block Modeling," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020.

[62] I. Abel and H. Graeb, "FUBOCO: Structure Synthesis of Basic Op-Amps by FUnctional BlOck COmposition," *ACM Trans. Des. Autom. Electron. Syst.*, Feb 2022, just Accepted. [Online]. Available: https://doi.org/10.1145/3522738

[63] I. Abel, C. Kowalsky, and H. Graeb, "A fast Structural Synthesis Algorithm for Op-Amps based on Multi-Threading Strategies," in *International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, 2021.

[64] M. Eick, "Structure and Signal Path Analysis for Analog and Digital Circuits," Ph.D. dissertation, Technische Universität München, 2013.

[65] M. Zwerger, "Verification and Synthesis of Analog Power-Down Circuits," Ph.D. dissertation, TU Munich, Jan 2017.

[66] A. Graupner, R. Jancke, and R. Wittmann, "Generator based approach for analog circuit and layout design and optimization," in *2011 Design, Automation Test in Europe*, 2011, pp. 1–6.

[67] T. McConaghy, P. Palmers, M. Steyaert, and G. G. E. Gielen, "Trustworthy Genetic Programming-Based Synthesis of Analog Circuit Topologies Using Hierarchical Domain-Specific Building Blocks," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 4, pp. 557–570, 2011.

[68] H. Li, F. Jiao, and A. Doboli, "Analog circuit topological feature extraction with unsupervised learning of new sub-structures," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2016, pp. 1509–1512.

[69] M. Neuner, I. Abel, and H. Graeb, "Library-free Structure Recognition for Analog Circuits," in *Design, Automation and Test in Europe (DATE))*, 2021.

[70] D. M. Binkley, C. E. Hopper, S. D. Tucker, B. C. Moss, J. M. Rochelle, and D. P. Foty, "A CAD methodology for optimizing transistor current and sizing in analog CMOS design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2003.

[71] H. Shichman and D. A. Hodges, "Modeling and simulation of insulated-gate field-effect transistor switching circuits," *IEEE Journal of Solid-State Circuits SC*, 1968.

[72] K. Antreich, H. Graeb, and C. Wieser, "Circuit analysis and optimization driven by worst-case distances," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1994.

[73] O. Cuate, O. Schuetze, F. Grasso, and E. Tlelo-Cuautle, "Sizing CMOS operational transconductance amplifiers applying NSGA-II and MOEAD," in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2019, pp. 149–152.

[74] *WiCkeD*, www.muneda.com, MunEDA, 2009. [Online]. Available: www.muneda.com

[75] Z. Rojec, r. Búrmen, and I. Fajfar, "An evolution-driven analog circuit topology synthesis," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016, pp. 1–6.

[76] Ka Nang Leung and P. K. T. Mok, "Analysis of multistage amplifier-frequency compensation," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, no. 9, 2001.

[77] J. Ramos, Xiaohong Peng, M. Steyaert, and W. Sansen, "Three stage amplifier frequency compensation," in *ESSCIRC 2004 - 29th European Solid-State Circuits Conference (IEEE Cat. No.03EX705)*, 2003.

[78] G. Palmisano and G. Palumbo, "An optimized compensation strategy for two-stage CMOS op amps," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 42, no. 3, 1995.

[79] ——, "A compensation strategy for two-stage CMOS opamps based on current buffer," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 44, no. 3, 1997.

[80] B. Razavi, *Design of Analog CMOS Integrated Circuits*. Tata McGraw-Hill, 2002.

[81] T. W. Francesca Rossi, Peter van Beek, *Handbook of Constraint Programming*, 2nd ed. Elsevier, 2006, vol. Volume 2.

[82] C. Schulte, G. Tack, and M. Z. Lagerkvist, "Modeling and Programming with Gecode," https://www.gecode.org/doc-latest/MPG.pdf, 2019.

[83] G. Tack, C. Schulte, and G. Smolka, "Generating propagators for finite set constraints," in *Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming*, F. Benhamou, Ed. Springer, 2006.

[84] T. Fruhwirth and S. Abdennadher, *Essentials of Constraint Programming*. Berlin, Heidelberg: Springer-Verlag, 2003.

[85] P. Hofstedt, *Multiparadigm Constraint Programming Languages*. Springer, 2011.

[86] I. P. Gent, C. Jefferson, and I. Miguel, "MINION: A Fast, Scalable, Constraint Solver," in *17th European Conference on Artificial Intelligence ECAI*, 2006.

[87] R. J. Dakin, "A tree-search algorithm for mixed integer programming problems," *The Computer Journal*, vol. 8, no. 3, pp. 250–255, 01 1965. [Online]. Available: https://doi.org/10.1093/comjnl/8.3.250

[88] W. D. Harvey, "Nonsystematic backtracking search," Ph.D. dissertation, Stanford University, 1995.

[89] M. Luby, A. Sinclair, and D. Zuckerman, "Optimal speedup of Las Vegas algorithms," in *Proceedings of the Second Isreal Symposium on the Theory of Computing and Systems*, 1993.

[90] A. K. Mackworth, "Consistency in networks of relations," in *Artificial Intelligence*, vol. 8, 1977, pp. 99–118.

[91] ——, "On reading sketch maps," in *5th International Joint Conference Artificial Intelligence, IJCAI*, vol. 8, 1977, pp. 598–606.

[92] M. Z. L. Christian Schulte, Guido Tack, *Modeling and Programming with Gecode*, 6th ed., www.gecode.org, May 2018.

[93] K. Wing-Hung, "IC Design of Power Management Circuits," in *IEEE International Symposium on Integrated Circuits, Tutorial*, 2009.

[94] P. T. Tran, H. L. Hess, and K. V. Noren, "Operational amplifier design with gain-enhancement differential amplifier," in *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, Oct 2012, pp. 6248–6253.

[95] R. E. Vallee and E. I. El-Masry, "A very high-frequency CMOS complementary folded cascode amplifier," in *IEEE Journal of Solid-State Circuits*, vol. 29, 1994, pp. 130–133.

[96] B. V. Amini and F. Ayazi, "Micro-gravity capacitive silicon-on-insulator accelerometers," in *Journal of Micromechanics and Microengineering*, vol. 15, no. 11. IOP Publishing, sep 2005, pp. 2113–2120.

[97] M. Banu, J. M. Khoury, and Y. Tsividis, "Fully differential operational amplifiers with accurate output balancing," in *IEEE Journal of Solid-State Circuits*, vol. 23, no. 6, Dec 1988, pp. 1410–1414.

[98] E. Tlelo-Cuautle, M. A. Valencia-Ponce, and L. G. de la Fraga, "Sizing CMOS Amplifiers by PSO and MOL to Improve DC Operating Point Conditions," *Electronics*, vol. 9, no. 6, 2020.

[99] G. Shi, "Automatic Stage-Form Circuit Reduction for Multistage Opamp Design Equation Generation," *ACM Transactions on Design Automation of Electronic Systems*, vol. 25, no. 1, Oct. 2019.

[100] ——, "Topological Approach to Symbolic Pole–Zero Extraction Incorporating Design Knowledge," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 11, 2017.