論文 / 著書情報
Article / Book Information

題目(和文)	トランジスタの簡易化モデルを用いたアナログ回路の自動設計
Title(English)	Automated design of analog circuits accelerated by use of simplified transistor model
著者(和文)	海野直之
Author(English)	NAOYUKI UNNO
出典(和文)	学位:博士（工学）, 学位授与機関:東京工業大学, 報告番号:甲第6821号, 授与年月日:2007年3月26日, 学位の種別:課程博士, 審査員:
Citation(English)	Degree:Doctor of Engineering, Conferring organization: Tokyo Institute of Technology, Report number:甲第6821号, Conferred date:2007/3/26, Degree Type:Course doctor, Examiner:
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Automated Design of Analog Circuits Accelerated by Use of Simplified Transistor Model

Naoyuki Unno

Submitted in partial fulfillment of
the requirements for the degree of
Doctor of Engineering

Tokyo Institute of Technology

2007

Acknowledgements

First of all, I have no words to express my deep gratitude to my supervisor, Professor Nobuo Fujii, for his support, guidance, patience and encouragement during this work. Without his guidance, I could not finish this work. Also, I would like to express my deep appreciation to my mentor, Associate Professor Eisuke Tokumitsu, for his precious advices and encouragement.

I would like to express my gratitude to Professor Akinori Nishihara, Professor Akira Matsuzawa and Professor Shigetaka Takagi from Tokyo Institute of Technology and Professor Toshiro Tsukada from STARC for sharing many valuable discussions and comments on my research.

Dr. Hajime Shibata from Analog Devices Inc. led me in this field of research. I would like to offer special thanks to him for his help and fruitful discussions, suggestions and encouragement.

I am really grateful to Dr. Takahide Sato, Assistant Professor of Fujii Laboratory and Dr. Nicodimus Retdian Agung Wahyu Wijaya, Assistant Professor of Takagi Laboratory for supporting me during my study for doctoral degree.

I should mention that Ms. Yoko Takashima who is a secretary of Fujii Laboratory, have helped me with a lot of paper procedures for my conference trips and publication of my papers. I am really indebted to her. I would also like to express my thanks to Tomomi Takagi who was a secretary of Nishihara Laboratory and Emi Muratsubaki who was a secretary of Takagi Laboratory for their help for the paperwork.

Mr. Chatree Budsabathon from Nishihara Laboratory is a person who have studied together during master course and doctoral course. I am glad to have worked with him. My friends from doctoral courses in Five Electronic Majors at Tokyo Institute of Technology, too many to mention their names, and Ms. Akane Saito from the Office of 21st Century COE Program, Photonics Nanodevice Integration Engineering, have enriched my doctoral life and given a lot of stimulation and encouragements through activities of Five Electronic Majors Doctoral Students Forum. It is a great pleasure for me to have shared four years during my study for doctoral

degree with them.

I also would like to say my thanks to all members of Fujii-Takagi Laboratory, Tsukada Laboratory, and Nishihara Laboratory for working hard together.

Finally, I would like to express my great thanks to my parents for giving me a chance to study in the doctoral course and their continuous support and encouragement and to my wife Pei-Ting Fu for her persistent help and encouragement.

Contents

Chapter 1

Introduction

1.1 Backgrounds

Since the invention of Integrated Circuit by Jack S. Kilby in 1958, increase of the number of transistors in an integrated circuit by downsizing of the transistor based on scaling law has been one of the most important technological trends. While the digital integrated circuits, especially microprocessors and memories, had been amazingly developed, analog integrated circuits had implemented simple functions such as amplifiers, mixers, converters, etc. until 1990s. However, the trend of RF-CMOS sprung out in mid-1990s has developed larger scale analog circuits into the integrated circuit[1]. In 2000s, markets of digital consumer electronics and vehicle electronics systems have been extremely increased and development of integrated circuit process technology has enabled system level integration including analog and digital signal processing systems on a single chip. For example, a DVD system which consists of more than 10 million transistors now can be integrated on a single chip[2]. Thus, SoC (Systems on a Chip) have become of major importance in LSI industry. The design of such complicated systems is a time-consuming task and computer-aided design (CAD) tools have inevitably been adopted to support automated design of such large scale integrated circuits.

In the digital domain, the design automation is well-developed and CAD tools are commercially available. This is owing to the nature of digital systems. A digital system can be described by Boolean expression and programming-language-like description, and its functionality can be expressed in an algorithmic form. Consequently, the design methodology has changed to more abstract description of a circuit. High-level synthesis tools synthesize the hardware description language (HDL) such as VHDL or Verilog at the behavior level or the structural level, then logic synthesis tools translate the HDL statement into a gate-level netlist. Layout tools

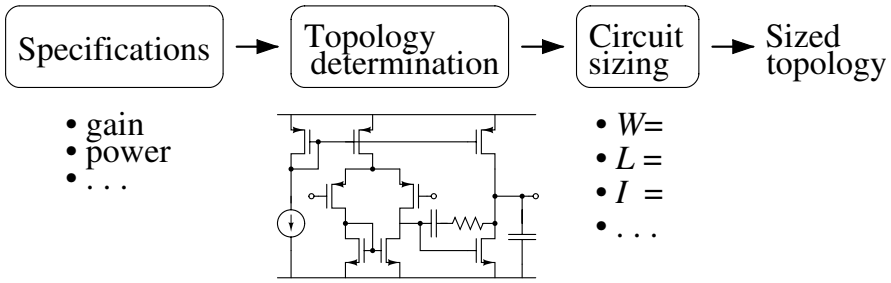


Figure 1.1: Basic process of analog circuit design.

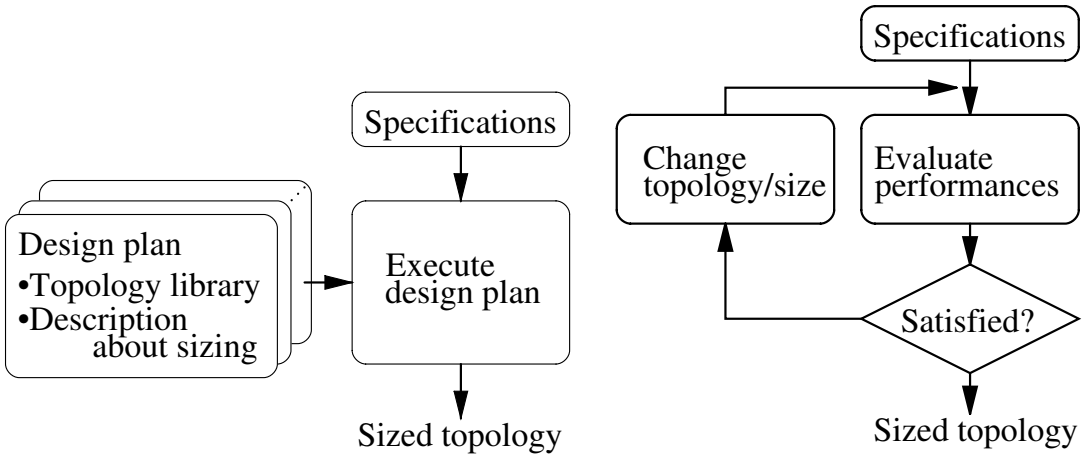
generate a mask-level layout from this netlist based on a cell library. Thus, the current digital circuit design is to describe a digital circuit by using a high-level description language.

By contrast, CAD tools for the analog circuit design available in the market had been circuit simulators such as SPICE and layout editing environments until the early 2000s. Although some tools for parameter optimization have appeared in the market for these few years, the synthesis of analog circuit topology is still in its infancy. Therefore, although 90% of the systems may comprise digital circuits, the design of analog circuits dominates more than 90% of the design time.

Some of the main reasons for this are that analog circuit functions are described by continuous functions and generally realized by using physical characteristics of semiconductor devices and that there are a lot of varieties of circuit schematics and conflicting specifications, and device sizes must be changed to corresponding to the schematics and specifications. Consequently, the analog circuit design methodology is not as straightforward as the digital circuit design. It is much more heuristic, intuitive, and requiring specialized design knowledge and skills acquired through a lot of experience.

The process of the analog circuit design is divided into two parts: determination of the circuit topology and sizing of the circuit components as shown in Fig. 1.1. Therefore, automated analog circuit synthesis systems cope with both of two parts or just circuit sizing for a given circuit topology.

A lot of automated analog circuit design methods have been proposed until now. They are basically classified to two approaches. One is exploiting analog design knowledge and heuristics (knowledge-based approach) and the other is using optimization techniques (optimization-based approach), whose brief flows are depicted in Fig. 1.2. The knowledge-based approach is a straightforward process and the optimization approach has iterations of change and evaluation. The optimization-based approach has some subcategories according to the method to evaluate cir-



(a) Knowledge-based (b) Optimization-based

Figure 1.2: Two basic approaches of analog circuit synthesis.

cuit performances, e.g. using symbolic models (equation-based approach) or using simulators(simulation-based approach).

In the beginning of analog circuit synthesis systems, knowledge-based methods treating both of topological and sizing issues have been proposed[3]–[6]. In these methods, the system selects one topology from a library of circuit topologies which were manually designed in advance by the human designers and carries out the circuit sizing by using a corresponding formal description of the circuits. To simplify the reuse of design knowledge and design task, a hierarchically structured frame work where circuit topologies were represented as an interconnection of sub-blocks was introduced in [4]. A more artificial intelligence approach by integration of both intuitive and formal knowledge in one program was adopted in [5]. These knowledge-based approaches require very time-consuming process to prepare design knowledge, i.e. manually designed circuit topologies and formal descriptions for the sizing. Besides, the design flexibility is limited to these prepared knowledge.

Although the synthesis systems in the early era dealt with both of the topological and sizing issues, the topological issue has received less attention and the sizing issue was mainly focused. Since the circuit sizing or optimization of component values is a multi-variable multi-objective numerical problem, great progress has been made in this issue. In the era when the computer power was poor, mainly equation-based methods have been reported. In [6], the circuit characteristics are described by analytic models and device sizes are optimized by a steepest descent algorithm. However, the analytic model for a new topology must be created by a

good analog designer. In [7] the analytic model is automatically generated by means of the symbolic simulator ISAAC[8] and circuit sizes are optimized by a simulated annealing. The symbolic simulator technique results in large expanded expression which restricts their usefulness for large circuits. The design of analog circuits can be formulated as posynomial convex problem and the global optimum can be solved by using a geometric programming in an extremely efficient way[9], [10]. The big drawback of this method is the analytic model must be posynomial function and not all design characteristics are easy to be expressed with sufficient accuracy. To solve the problem of automatic generation of accurate models, neural network models are used in [11]. However, every time the circuit topology is changed, the neural network models must be trained by using data generated from SPICE simulations.

As the computer power was improved, simulation-based methods have been practical. The main difference of simulation-based methods is the optimization algorithm. Some methods used a single algorithm such as nonlinear optimization technique[12], [13], simulated annealing[14], [15], genetic algorithm[16]–[18], immune algorithm[19] and particle swarm optimization[20], and other employed the combination of plural algorithms such as a combined annealing-genetic algorithm[21], particle swarm optimization and adaptive simulated annealing with tunneling[22], evolutionary programming combined with a gradient-based search method[23]. A stochastic pattern search adopted in [24] can be easily distributed over a pool of workstations. Some tools are now commercially available[25]. More discussion of the circuit sizing is detailed in [26].

Since the enormous effort has been solving the circuit sizing issue, the topological issue is more important. Some topology selection methods based on the optimization-based approach have been proposed [27], [28]. However, the topology selection method has the same disadvantage as the knowledge-based approach that generated circuit topologies are limited to the circuit structures prepared by the human engineers.

Koza et al. proposed the synthesis by means of genetic programming[29]. This method can generate a circuit by evolving both of circuit topologies and sizes simultaneously without any design knowledge and initial circuit. The problem of this method is that it requires a huge computing time and generated circuits are far from well-known circuit structures and hard to understand. This causes a big problem in the fabrication of the circuits because the designers cannot amend the circuits if the fabricated circuits do not operate properly as they are designed.

In the other hand, Grimbleby proposed the system which generated circuit topologies by using genetic algorithm and sized them by the symbolic analysis and

quasi-Newton algorithm[30]. Although it can generate a more reasonable circuit in less time, it can be applied to only the linear circuit because the symbolic analysis can be used for linear characteristics.

Several methods exploit design knowledge in the circuit representation scheme which restricts the circuit structure in order to reduce the search space so that the synthesis time is reduced and more reasonable circuits can be generated. Lohn et al. proposed a linear representation[31] and Hou et al. proposed a tree representation[32], which simply express series or shunt connections in a circuit. However, these capability to express analog circuit is limited. Shibata et al. proposed a coding based on current path [33], [34] and Sripramong et al. proposed a circuit representation technique based on current-flow analysis[35]. Circuits generated by these methods still contain many strange and meaningless connections of components. Natsui et al. applied the graph representation which can directly manipulate the circuit graphs without encoding their structures into bit strings or other indirect representations[36]. This requires a number of simulations even when it only generates a current mirror circuit.

Methods based on topological reuse have been proposed to speed up the synthesis [37]–[39]. These methods reuse sub-circuits which are well-known, generated by dividing given prototype circuits, or obtained during the synthesis. However, criteria for reuse is based only on topology features and is not meaningful to specifications. Therefore, these methods still tend to be time-consuming.

Another approach is evolvable hardware or reconfigurable hardware optimized by evolutionary computation[40], [41]. This approach uses real hardware whose actual output response is evaluated in the evolutionary computation process. Stoica et al. proposed a reconfigurable VLSI architecture that is called field programmable transistor array(FPTA)[42], [43]. Although this approach solves the realization problem of automated design circuits, it requires a microprocessor, analog-to-digital and digital-to-analog converters to evaluate the circuit. When the reconfigurable feature is not necessary, such extra circuits may cost higher.

So far the computer-aided design of analog circuits has been outlined. From the discussion, the computer-aided design tools are inevitable to support the design of large-scale analog circuits. Now that the circuit sizing tools are commercially available, the automated synthesis of analog circuit topology is the most important issue. As referred to above, the heuristic method such as genetic algorithm seems a promising approach to address this problem. However, it requires a huge computing time and sometimes any circuit that meets a given specification may not be obtained. If a circuit is accidentally obtained, it might be quite complicated and contain many

strange and meaningless connections of circuit components.

1.2 Motivation and Objectives

This dissertation describes the synthesis of analog circuit topology where a satisfactory circuit can be generated without fail in a short computing time. The work can be divided into two parts. In the first part, the approach is to shrink the search space reasonably without limiting the circuit structure. By shrinking the search space, it is expected to reduce the synthesis time. For this purpose, simplified models of transistor at the first stage of the topology synthesis are introduced.

In the second part, the aim is to accelerate the evolutionary process. Since there are a lot of fruitless genetic operations during usual evolutionary processes, reduction of the fruitless operations must result in the speed up the synthesis and the reduction of the design time. To achieve that, the genetic algorithm with reuse of genetic operations is introduced.

1.3 Organization of the Dissertation

In Chapter 2, the basics of automated design by evolutionary computation are reviewed. Definitions and essences are discussed in this chapter.

In Chapter 3, synthesis starting with simplified elements is proposed. After introducing the simplified elements, the synthesis algorithm using simplified elements and methods to replace simplified elements by MOSFETs are explained. Its capability is demonstrated through experiments of synthesis of linear and nonlinear analog circuits and benchmark tests.

In Chapter 4, synthesis with reuse of genetic operations is proposed. Methods and criteria for store and reuse of genetic operations are explained and effectiveness of the reuse is demonstrated by three design examples which are quite difficult to design even by skilled analog circuit engineers. The conclusions are described in Chapter 5.

Chapter 2

Review of Automated Design by Evolutionary Computation

2.1 Introduction

Evolutionary computation, which emulates the evolutionary process in the natural world, is a powerful optimization algorithm and a promising approach to address the automated design of analog circuit topology. In this chapter, the overview of the evolutionary computation is explained, then an automated analog circuit design system based on a genetic algorithm is introduced. In subsequent chapters, analog circuits are generated in the same process as explained in this chapter. Also, coding scheme and fitness functions which are important issues in the genetic algorithm are discussed in the following sections.

2.2 Evolutionary Computation

2.2.1 Overview

The evolutionary computation is a multi-point search algorithm whose concept is survival of fittest. A search point is called individual and the individual has its genetic code called genotype or genome. The genome is decoded to an actual solution candidate in the search space, which is called phenotype in genetic biological terms, and its performance is evaluated. Then, the numerical index called fitness is calculated from the performance. The calculating formula is called fitness function, which is defined in advance by a user. Good individuals are selected according to the fitness from the population in a generation and new individuals in the next generation are reproduced by genetic operations. As generations pass by, individuals

are evolved so that their phenotype performance reaches to the required one.

The relationship between the genome and fitness is shown in Fig. 2.1, with the case where the evolutionary computation is applied to analog circuit design and the case of numerical computation for comparison. While the evaluation value is directly

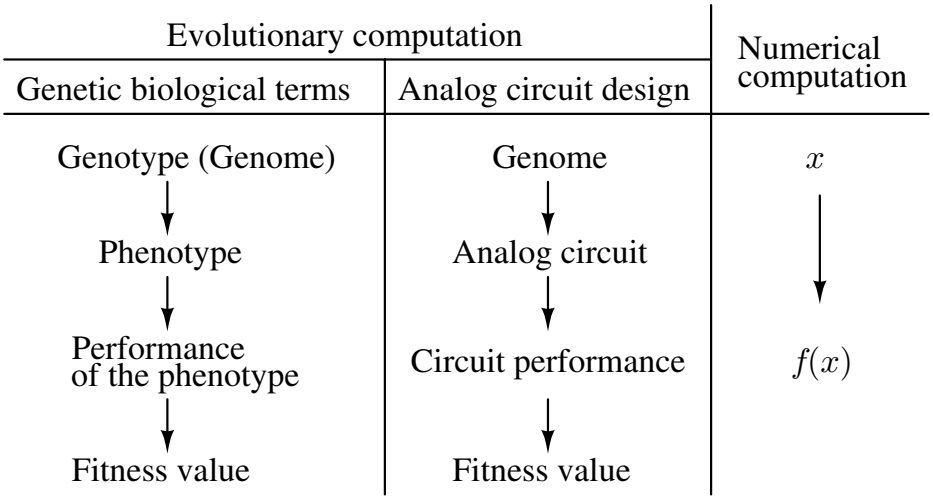


Figure 2.1: Evolutionary and numerical computation.

calculated from a solution candidate x by an objective function $f(x)$ in the numerical computation, two conversions, one is from the genotype to the phenotype and the other is from the performance to fitness value, are necessary in the evolutionary computation. For example, the conversions from the genome to analog circuit and from circuit performances to fitness value are necessary in the analog circuit design. However, the evolutionary computation can be applied to any problem when solution candidates of the problem to be solved are represented by the genomes and their fitness values can be calculated by their performances. Since the search by the evolutionary computation is supposed to rarely land in a local optimum and find a feasible solution even when it is hard to solve the problem theoretically or by a deterministic algorithm, the evolutionary computation is applied for many kinds of problems.

Currently, the evolutionary computation is classified into four types of algorithms: Evolutionary Strategy, Evolutionary Programming, Genetic Algorithm and Genetic Programming. However, the main differences of them are genetic representation and genetic operation to be used to change genomes, and the process flow is almost the same. Therefore, the discussion focuses on the genetic algorithm from here.

The genetic algorithm generates search points by two types of genetic operations,

Crossover and Mutation. Crossover generates two Offspring from two Parents by exchanging parts of the two genomes and Mutation generates an Offspring from a Parent by change a part of the genome. Therefore, it is supposed to excel at global search, but be weak in local search around a good solution. A genetic algorithm combined with a local search method (Genetic Local Search) redeems this weak point. In this algorithm, the local search seeks a better solution in the neighborhood of an individual when the individual is evaluated. Its effectiveness is discussed in [44].

2.2.2 Coding Scheme

When the genetic algorithm is used to solve a problem, the coding scheme by which a genome is decoded to the corresponding solution candidate is an important issue to successfully solve the problem. Features required for a good coding are to effectively cover solution candidates in all over the search space with uniqueness, continuity and simplicity. Uniqueness is necessary because if some different genomes are decoded to the same solution, the genetic representation is redundant and the search process requires more computing time. At the same time, similar solutions have to be expressed by similar genomes. Otherwise, a small change in a genome by a genetic operation causes a big change in its phenotype and the algorithm becomes more random one. Simplicity is needed to easily manipulate genomes by genetic operations.

2.2.3 Fitness Function

The fitness function is used to calculate the fitness from the performances of a solution candidate. Since the concept of evolutionary computation is survival of fittest, the fitness function must reflect how close the solution candidate is to a feasible solution. When the larger fitness is closer to the feasible solution, a genome having larger fitness must have more chances to be selected for the next generation. Since the definition of the fitness function heavily affects the performance of the genetic algorithm depending on the applications and specifications, the fitness function must be defined carefully.

2.3 Automated Analog Circuit Design by Evolutionary Computation

When the evolutionary computation based on the genetic algorithm is used for the automated analog circuit design, it must be considered to evolve the circuit topology and component values simultaneously or separately. While the circuit topology and component values are handled separately in [30], most of reported methods handle them simultaneously because the search of the best component values for a given circuit topology is neither easy nor short-time task. However, component values severely affect performances of the analog circuit. Well-adjusted component values bring out good circuit performances and vice versa. Therefore, simultaneous handling of the circuit topology and component values sometimes misses a good circuit structure because its component values are not well-adjusted and it results in generating complicated circuits. Thus, the local search or optimization of component values is necessary after a circuit topology is generated. [30] indicates the importance of the optimization of component values. However, some idea must be installed so that component values are easily adjusted well.

The evolutionary computation based on the genetic algorithm combined with local search is used to create circuits in subsequence chapters. The genetic algorithm evolves circuit topologies and the local search optimizes component values. The synthesis flow is shown in Fig. 2.2. The fitness function is defined by a user before the synthesis starts. After initial genomes are randomly generated, every genome is converted to its corresponding analog circuit. The local search seeks the best design parameters by iterating three steps, changing parameters, simulation of the circuits and calculation of the fitness value from the simulation results. Then, individuals are selected according to the fitness and modified by genetic operations and new genomes in the next generation are reproduced. The four steps, i.e. local search, selection, genetic operation and reproduction are iterated until all the given specifications are satisfied.

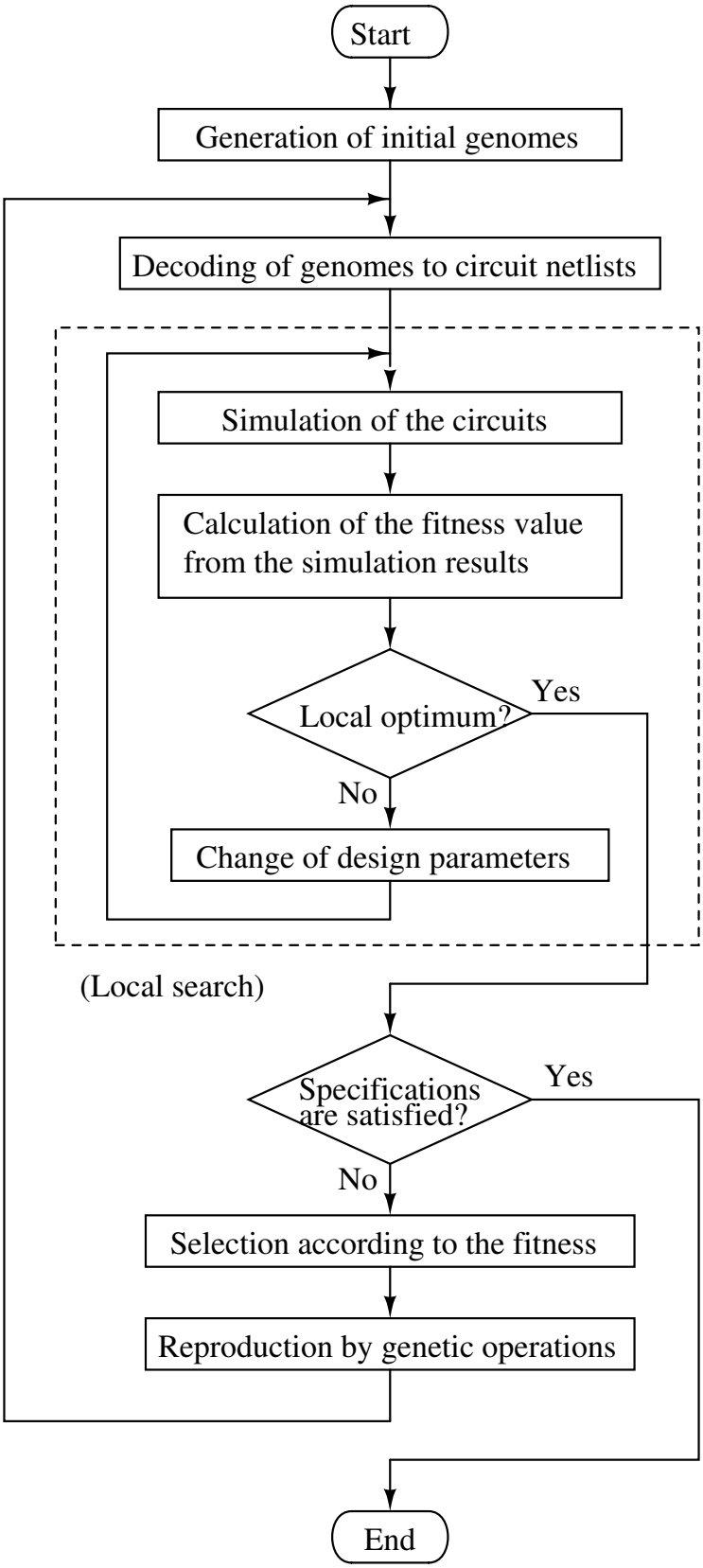


Figure 2.2: The flow of genetic algorithm with local search.

Chapter 3

Automated Design of Analog Circuits Starting with Simplified Elements

3.1 Introduction

This chapter proposes an automated design of analog circuits starting with simplified elements to reduce the design time. Instead of directly creating analog circuits with complicated transistor models, simplified models of transistor are introduced at the first stage of the topology synthesis. Since simplified elements operate without bias voltage or current, the search space can be shrunk without limiting the circuit structure. After a circuit topology is obtained using simplified elements, the simplified elements are replaced by transistors and the transistor parameters are optimized to meet the given specifications. This idea is based on a design process of skilled analog circuit designers. Firstly the analog circuit designers generally create a circuit assuming that transistors are expressed by simple ideal model such as a voltage controlled current source, and then they replace the ideal elements by practical transistors. The obtained circuits only contain transistors that operate in active region and consequently the circuits become quite reasonable.

In order to make it possible to design a circuit with noise consideration, which human can hardly treat, noise models for the simplified element are introduced. Thus we can specify noise figure, input referred noise, etc. Also, a paired simplified model of transistor is introduced so that differential circuits can effectively be synthesized.

The capability of this method is demonstrated through experiments of synthesis of a trans-impedance amplifier and a cubing circuit and benchmark tests. The

Table 3.1: Explanation of symbols.

C_{OX}	: Capacitance of gate oxide film
μ	: Carrier mobility
W	: Channel width
L	: Channel length
K	: Transconductance parameter ($= \mu C_{OX} W/L$)
λ	: Channel length modulation factor
V_{TH}	: Threshold voltage
K_f	: Process-dependent constant for the flicker noise
γ	: Process-dependent constant for the thermal noise
f	: Frequency
k	: Boltzmann constant
T	: Absolute temperature

design examples show that the proposed CAD is able to generate a reasonable circuit flexibly according to the given specifications and the results of the benchmark tests show that the proposed CAD is more than 10 times faster than the previous one that does not use simplified elements. Although this chapter concentrates the discussion on CMOS analog circuits, the proposed method can be applied to bipolar transistor circuits.

Table 3.1 shows symbols used in Chapter 3 and Chapter 4. The subscripts n and p attached to these symbols stand for n-MOSFET and p-MOSFET, respectively.

3.2 Simplified Signal Model of Transistor

3.2.1 Small Signal Linear Model of MOSFET

For the design of linear circuits such as amplifiers, trans-impedance amplifiers, etc., we basically require linear elements. Fig. 3.1 shows the simplified small signal linear model of MOSFET, where g_m is the transconductance and r_d is the output resistance of the MOSFET. Since the drain current I_D of MOSFET including the channel length modulation in the saturation region is given as

$$I_D = \frac{K}{2}(V_{GS} - V_{TH})^2(1 + \lambda V_{DS}) \tag{3.1}$$

where V_{GS} is the gate-source voltage and V_{DS} is the drain-source voltage. In the simplified model, neglecting the channel length modulation, g_m is given by differen-

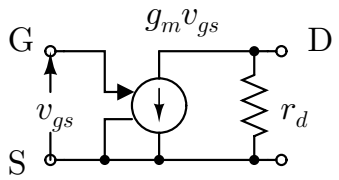


Figure 3.1: Small signal linear model of MOSFET.

tiating I_D with respect to V_{GS} as

$$g_m = \frac{\partial I_D}{\partial V_{GS}} = K(V_{GS} - V_{TH}). \tag{3.2}$$

r_d is given by inverse of differentiation of I_D with respect to V_{DS} as

$$r_d = \left(\frac{\partial I_D}{\partial V_{DS}}\right)^{-1} = \frac{1}{\lambda I_D}. \tag{3.3}$$

r_d is a nonlinear resistance depending on I_D, however, in the simplified model, we set r_d to a specified constant value in order to reduce design parameters. Thus we only have one design parameter, g_m. n and p channel MOSFETs have the same linear model given by Fig. 3.1.

3.2.2 Large Signal Model (Nonlinear Model) of MOSFET

In order to realize nonlinear characteristics, nonlinear devices are required. MOSFET is an intrinsically nonlinear device. The drain current I_D of n channel MOSFET (n-MOSFET) in the saturation region without the channel length modulation is given as

$$I_D = \frac{K_n}{2}(V_{GS} - V_{THN})^2. \tag{3.4}$$

We assume V_{THN} is a constant given by the IC process used. V_{GS} can be expressed as,

$$V_{GS} = v_{gs} + V_{GSQ} \tag{3.5}$$

where v_{gs} is a signal component and V_{GSQ} is a kind of the operating point. Substituting Eq. (3.5) into (3.4), we obtain,

$$\begin{aligned} I_D &= \frac{K_n}{2}(v_{gs} + V_{GSQ} - V_{THN})^2 \\ &= \frac{K_n}{2}v_{gs}^2 + K_n(V_{GSQ} - V_{THN})v_{gs} + \frac{K_n}{2}(V_{GSQ} - V_{THN})^2. \end{aligned} \tag{3.6}$$

The third term of Eq. (3.6) is a DC current which is a kind of the operating point of I_D. We denote this DC current as I_{DQ},

$$I_{DQ} = \frac{K_n}{2}(V_{GSQ} - V_{THN})^2. \tag{3.7}$$

Thus, the signal component i_d of I_D is given by

$$\begin{aligned} i_d &= I_D - I_{DQ} \\ &= \frac{K_n}{2}v_{gs}^2 + K_n(V_{GSQ} - V_{THN})v_{gs} \\ &= \alpha_n v_{gs}^2 + g_{mn}v_{gs} \end{aligned} \tag{3.8}$$

where $\alpha_n = \dfrac{K_n}{2}$ and g_{mn} is the transconductance of n-MOSFET at a current of $I_D = I_{DQ}$ which is given by

$$g_{mn} = K_n(V_{GSQ} - V_{THN}). \tag{3.9}$$

Eq. (3.8) shows that the nonlinear model of n-MOSFET is given by Fig. 3.2(a) where r_{dn} is an output resistance at $I_D = I_{DQ}$. When g_{mn} and α_n are given, r_{dn} is obtained by the following equation,

$$r_{dn} = \frac{I_{D0}}{I_{DQ}}r_{dn0} \tag{3.10}$$

where r_{dn0} is the drain resistance at $I_D = I_{D0}$. Although r_{dn} is a nonlinear resistance depending on I_D, we assume r_{dn} is a constant obtained by Eq. (3.10) in the simplified model. We also assume α_n is a constant to reduce the number of design parameters. Thus, finally we only have one design parameter, g_{mn} in the nonlinear model of Fig. 3.2(a).

For the case of nonlinear model, we must consider the difference between n and p channel MOSFETs. The drain current of p channel MOSFET(p-MOSFET) is given by

$$I_D = -\frac{K_p}{2}(-V_{GS} + V_{THP})^2. \tag{3.11}$$

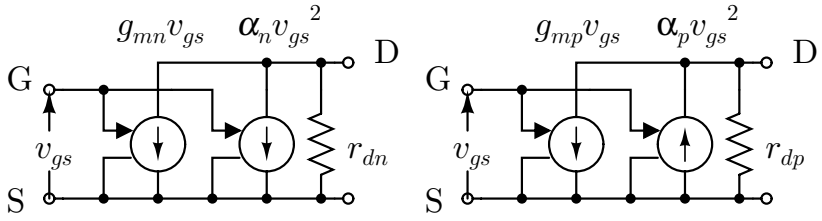
Similarly with n-MOSFET, we have

$$\begin{aligned} i_d &= I_D - I_{DQ} \\ &= -\alpha_p v_{gs}^2 + g_{mp}v_{gs}, \end{aligned} \tag{3.12}$$

where $\alpha_p = \dfrac{K_p}{2}$, and

$$I_{DQ} = -\frac{K_p}{2}(-V_{GSQ} + V_{THP})^2, \tag{3.13}$$

$$g_{mp} = K_p(-V_{GSQ} + V_{THP}). \tag{3.14}$$



(a) n-MOSFET (b) p-MOSFET

Figure 3.2: Nonlinear model of MOSFET.

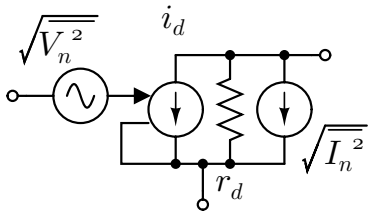


Figure 3.3: Noise model

From Eq. (3.12), we obtain a simplified nonlinear model of p-MOSFET as shown Fig. 3.2(b). r_{dp} is given from g_{mp} and α_p as,

$$r_{dp} = \frac{I_{D0}}{I_{DQ}} r_{dp0} \tag{3.15}$$

where r_{dp0} is the drain resistance at $I_D = I_{D0}$.

α_n and α_p determine the nonlinearity of MOSFETs and larger values of α_n and α_p give stronger nonlinear characteristics. We must select an appropriate value of α_n and α_p depending on the nonlinearity of the circuit to be designed. When $\alpha_n = \alpha_p = 0$, the nonlinear models of Figs. 3.2(a) and (b) coincide with the linear model of Fig. 3.1.

3.2.3 Noise Model for Simplified Signal Model of MOSFET

We add, if necessary, noise sources to each element as shown in Fig. 3.3 where $\overline{V_n^2}$ and $\overline{I_n^2}$ are the flicker noise and thermal one, respectively. Their spectral densities are given by [45]

$$\overline{V_n^2} = \frac{K_f}{WLC_{OX}f} \tag{3.16}$$

$$\overline{I_n^2} = 4kT\gamma g_m. \tag{3.17}$$

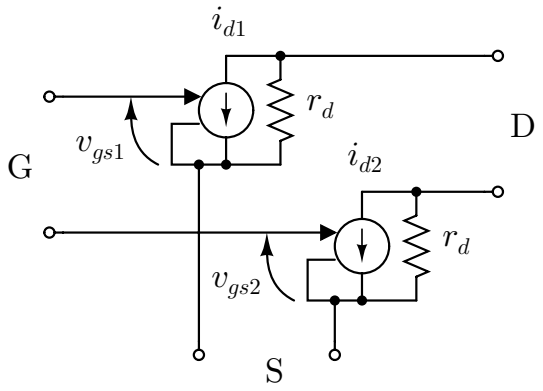


Figure 3.4: Paired simplified signal model of MOSFET

Here, multiplying Eqs. (3.2) and (3.3), we obtain,

$$g_m^2 r_d = \frac{2K}{\lambda} = \frac{2\mu C_{OX} W}{\lambda L}. \tag{3.18}$$

Since g_m^2 is proportional to W for given r_d and L, Eq. (3.16) can be expressed by using g_m as follows.

$$\overline{V_n^2} = \frac{K_f}{W_0 L_0 \left(\frac{g_m^2}{g_{m0}^2}\right) C_{OX} f} \tag{3.19}$$

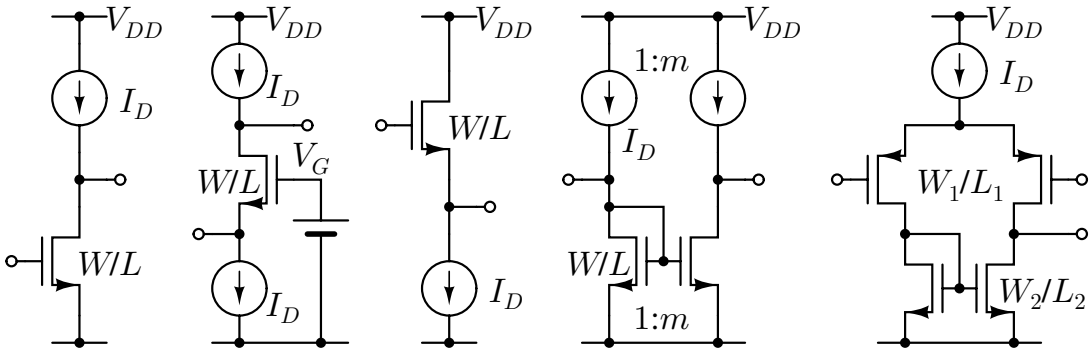
where W_0 and L_0 are the channel width and length when $g_m = g_{m0}$ and $r_d = r_{d0}$. From Eqs. (3.17) and (3.19), we find that we only have one design parameter g_m even when we consider noise performances. I_{D0}, r_{d0}, W_0, L_0 and g_{m0} can be obtained from sampled data of a MOSFET under a given MOS technology.

3.2.4 Paired Simplified Signal Model of MOSFET

In the design of differential circuits, symmetry of the circuit is inevitable. Therefore, in order to cope with differential circuit structures, the simplified model of MOSFET is used in pairs as shown in Fig. 3.4 where i_d is the drain signal current of the MOSFETs. i_d is given by the following equations as derived in Section 3.2.2,

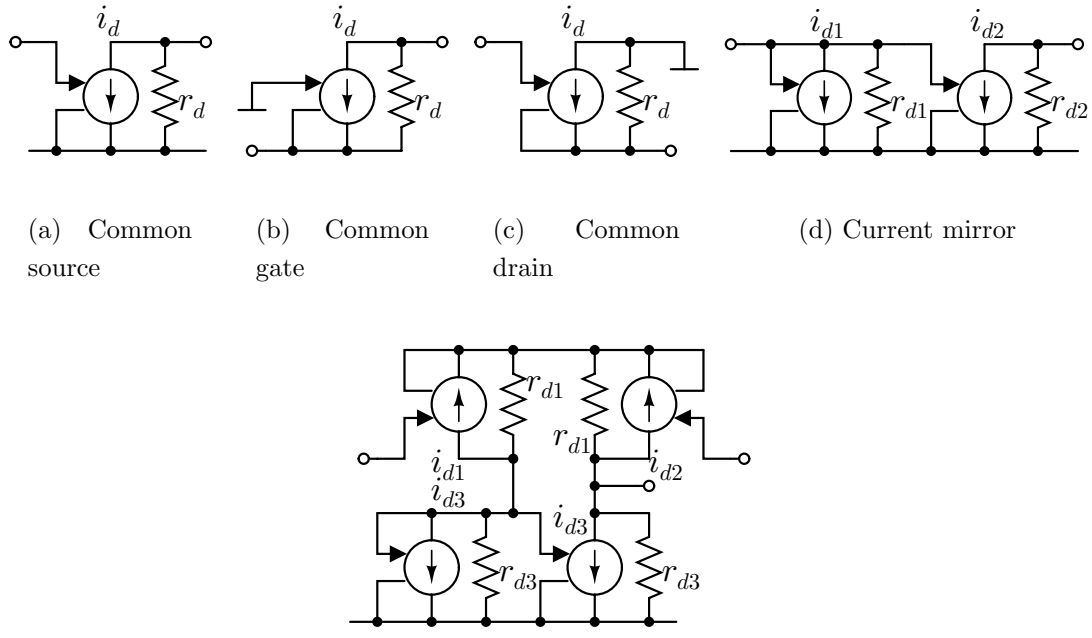
$$i_d = \begin{cases} \alpha_n v_{gs}^2 + g_{mn} v_{gs} & \text{for n-MOSFET,} \\ -\alpha_p v_{gs}^2 + g_{mp} v_{gs} & \text{for p-MOSFET,} \end{cases} \tag{3.20}$$

where $\alpha_n = \dfrac{K_n}{2}, \alpha_p = \dfrac{K_p}{2}$, and g_{mn} and g_{mnp} are given by Eq. (3.9) and (3.14), respectively. r_d is given by Eq. (3.10) or (3.15). Even in the design of the differential circuits, we only have one design parameter, g_m. For the design of differential linear circuits, we set $\alpha = 0$ and r_d can have an appropriate constant value.



(a) Common source (b) Common gate (c) Common drain (d) Current mirror (e) Differential amplifier

Figure 3.5: Basic MOSFET circuits.



(a) Common source (b) Common gate (c) Common drain (d) Current mirror

(e) Differential amplifier

Figure 3.6: Circuits with simplified element equivalent to Fig. 3.5.

3.2.5 Reduction Ratio of the Search Space

This section explains how the search space of the synthesis is shrunk by introducing the simplified elements. There are two point of views. One is the number of elements

and the other is the number of design parameters. To explain from the point of view of the number of elements, consider five basic MOSFET circuits shown in Fig. 3.5 and circuits with simplified elements shown in Fig. 3.6 which are equivalent to Fig. 3.5. In Fig. 3.5, MOSFETs of the bias current sources are expressed by ideal current sources for the simplicity of the circuit diagram. Since analog circuits consist of these basic circuits, we can see from these figures that the number of transistors which are necessary to compose analog circuits is reduced by 53% in average by using simplified elements. Since the search space of circuit topology increases exponentially as the number of elements increases, it is shrunk exponentially by introducing simplified elements.

To see from the point of view of the number of design parameters, consider the circuit of Fig. 3.5(c) and Fig. 3.6(c) for example. Assuming the MOSFET size can take 10 discrete values, combination number of the sizes of two MOSFETs is 100. However, some of the combinations make the circuit operate in deep linear or cut-off region and others make the circuit have the same g_m. Since circuit performances are basically determined by g_m, such combinations are useless and redundant. On the other hand, the simplified elements operate in saturation region with a given g_m. When there are 10 discrete g_m, there are 10 different performances and there is no useless or redundant parameter. Besides, MOSFET circuits are affected by other connected circuits and this generates more useless or redundant parameter combinations. In contrast, circuits with simplified elements keep the given g_m regardless of other connected elements. Therefore, the search space is also greatly reduced from the point of the number of design parameters.

3.3 Synthesis Using Simplified Elements

3.3.1 Genetic Representation

Since neither bias voltage nor current are necessary for simplified elements, only signal paths of the circuit have be considered during the synthesis. Therefore, a graph representation with respect to the signal path is employed in order to express analog circuits without encoding circuits indirect genetic representations such as bit strings. The graphical treatment is similar to [36] and an example of the circuit graph is shown in Fig. 3.7. A circuit graph consists of four items depicted in Fig. 3.8. In Fig. 3.8, T_1 and T_2 mean two terminals or terminal pairs of connected simplified elements and the arrow expresses the connection and signal transfer direction between two elements. In the synthesis of differential circuits, cross-coupled connection is allowed when two pairs are connected. Also, source terminals of two elements in

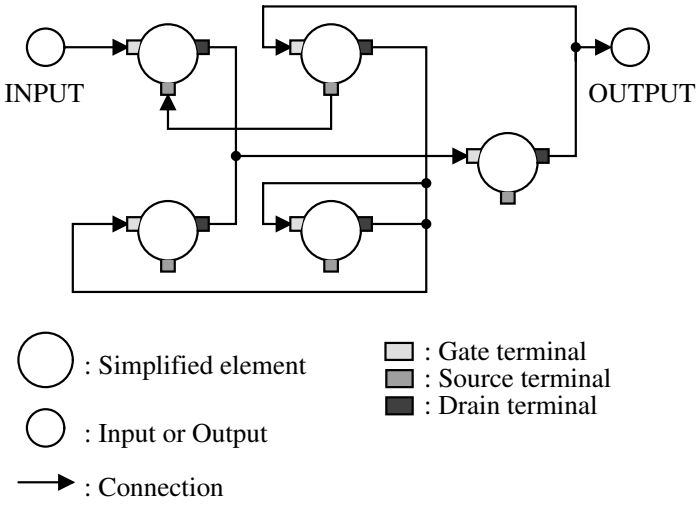


Figure 3.7: A circuit graph.

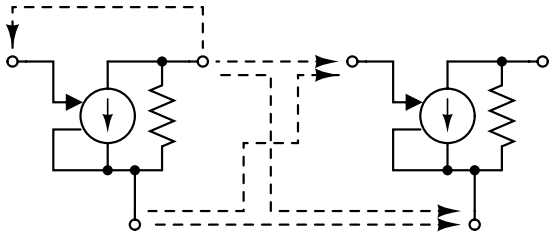
	For single-phase circuits		For differential circuits	
Name	Symbol	Schematic	Symbol	Schematic
Input node	IN	v_{in} or i_{in}	IN	v_{in1} v_{in2} or i_{in1} i_{in2}
Output node	OUT	v_{out} or i_{out}	OUT	v_{out1} v_{out2} or i_{out1} i_{out2}
Simplified element				
Connection	T_1 T_2	T_1 T_2	T_1 T_2	T_1 T_2

Figure 3.8: Symbols in the circuit graph.

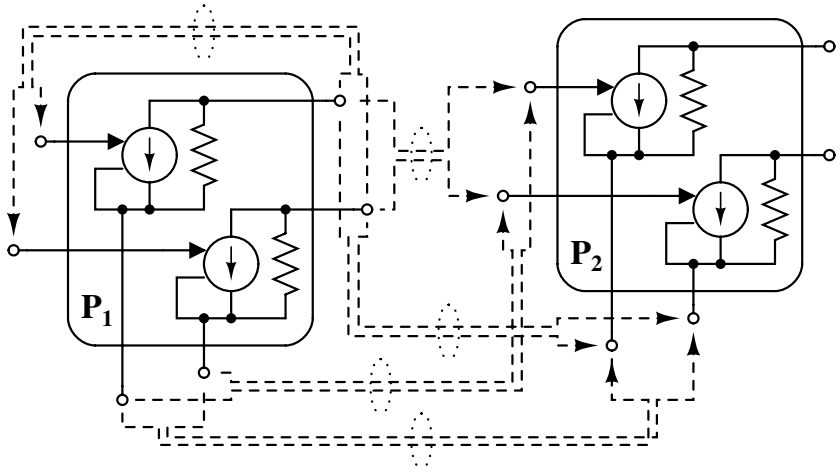
each pair can be shorted to form a source-coupled pair. Throughout the synthesis, the connection of the terminals of transistor is restricted to the five cases shown in Fig. 3.9 to avoid meaningless connections taking the signal flows into consideration. When a terminal of transistor is not connected to any other terminal, it is grounded.

3.3.2 Genetic Operation

Since the simplified models given by Figs. 3.1, 3.2(a) and (b) are only available for the signal, we must only use the specifications which are defined by signals such



(a) Single

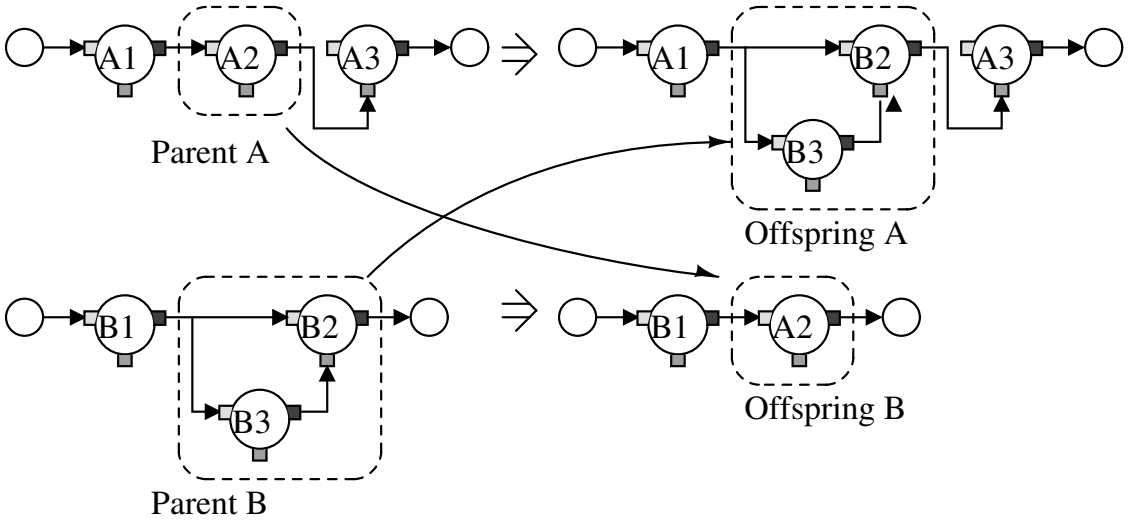


(b) Paired

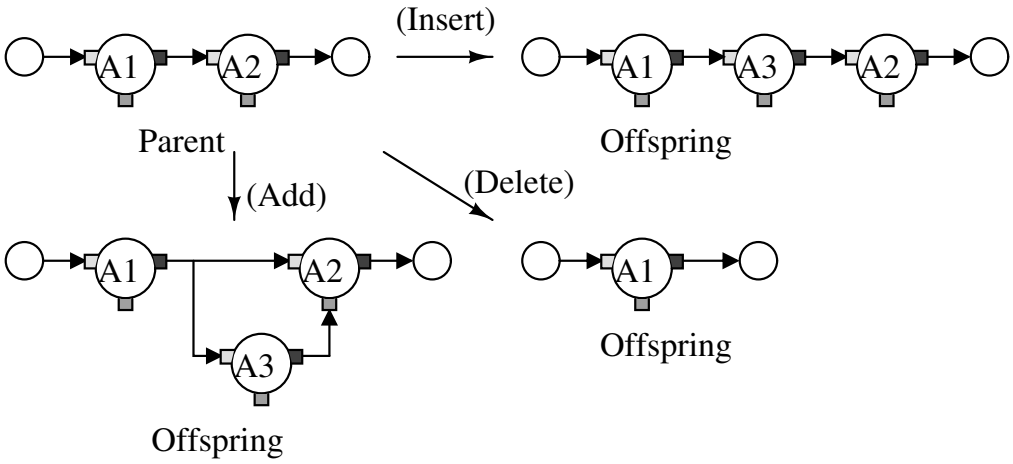
Figure 3.9: Connections of simplified elements.

as gain of amplifiers, input impedance, etc. Other specifications such as DC power consumption, area of the circuit, etc. are checked through the optimization process after the creation of the circuit.

The evolutionary computation based on the genetic algorithm combined with local search explained in Chapter 2 is used to create circuits with simplified elements. At the local search stage, g_m of each topology are changed to search the best g_m. Since g_m is the only variable parameter at this stage and every simplified element operates in active region, the local search can be done much more efficiently than that of MOSFET circuits. Next-Ascent Hill Climbing [46] is used as a local search method and BestN selection method [47] is used to select individuals. At the genetic operation stage, two types of genetic operation, Crossover and Mutation are used to reproduce new genomes in the next generation. Crossover exchanges subgraphs between two randomly selected genomes as shown in Fig. 3.10(a) where the two new genomes, Offspring A and B are reproduced from the two parents A and B



(a) Crossover



(b) Mutation

Figure 3.10: Genetic operations.

by exchanging the parts encircled by the dotted lines. Mutation inserts, adds, or deletes an element as shown in Fig. 3.10(b).

3.3.3 Fitness and Fitness Function

At the selection stage, we evaluate circuits by fitness functions. The fitness function of each specification can be defined by users. In this chapter, all the fitness functions

f_i are normalized to have a value between 0 and 1 as,

$$0 \leq f_i \leq 1. \tag{3.21}$$

When a specification is satisfied, the fitness function of the specification becomes 1. The total performance of a circuit is evaluated by the fitness F which is defined by

$$F = \sum_{i=1}^{N} f_i \tag{3.22}$$

where N is the number of specifications. Thus, all the given specifications are satisfied, then $F = N$.

3.4 Replacement of Simplified Elements with MOSFETs

3.4.1 Replacement of Small Signal Element

All the simplified elements in a circuit obtained by the synthesis must be replaced by MOSFETs. In order to convert a simplified transistor to a MOSFET, we need to calculate V_{GSQ}, I_{DQ}, and K. These values are obtained by the following two equations,

$$g_m r_d = \frac{2}{\lambda(V_{GSQ} - V_{TH})} \tag{3.23}$$

and

$$g_m = K(V_{GSQ} - V_{TH}) = \frac{2I_{DQ}}{V_{GSQ} - V_{TH}}, \tag{3.24}$$

where λ is the channel length modulation factor which is assumed to be a constant at this stage. First, V_{GSQ} is calculated from Eq. (3.23) and then, I_{DQ} and K can be calculated by g_m and V_{GSQ} from Eq. (3.24).

Since there is no distinction between n-MOSFET and p-MOSFET in the small signal linear model, the polarity of each simplified element has to be selected. When a circuit consists of N simplified elements, there are 2^N polarity combinations of MOSFETs. A method to select the polarity is shown in Fig. 3.11. As an example, we consider a circuit shown in Fig. 3.11(a) where two transistors M_1 and M_2 are simplified. First, we replace the simplified transistor M_1, to which the input signal is applied, with an n-MOSFET or a p-MOSFET as shown in Fig. 3.11(b). Then,

the performances of these two circuits are compared by simulation and the superior one is selected as the first stage. Next, the second element M_2 is replaced by an n-MOSFET or a p-MOSFET as shown in Fig. 3.11(c) where we assume that the n-channel first stage has been selected. These two circuits are similarly compared by simulation. This process is repeated until all the simplified elements are replaced by MOSFETs. $2N$ simulations are required to replace all the N simplified elements with MOSFETs.

There may be some cases in which an obtained polarity combination depends on the replacement order. But if simplified elements are properly replaced with MOSFETs having the same g_m and r_d as simplified elements, specifications defined by signals are necessarily satisfied.

3.4.2 Replacement of Large Signal Element

Replacement

Since $\alpha(= K/2)$ in Eqs. (3.8) and (3.12) is constant, V_{GSQ} is calculated from Eqs. (3.9) or (3.14) as follows,

$$V_{GSQ} = \frac{g_{mn}}{K_n} + V_{THN}, \tag{3.25}$$

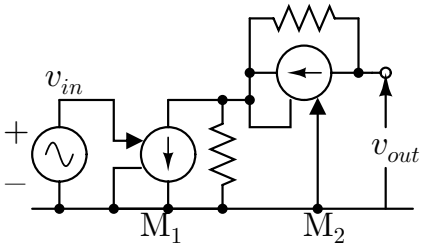
or

$$V_{GSQ} = -\frac{g_{mp}}{K_p} + V_{THP}, \tag{3.26}$$

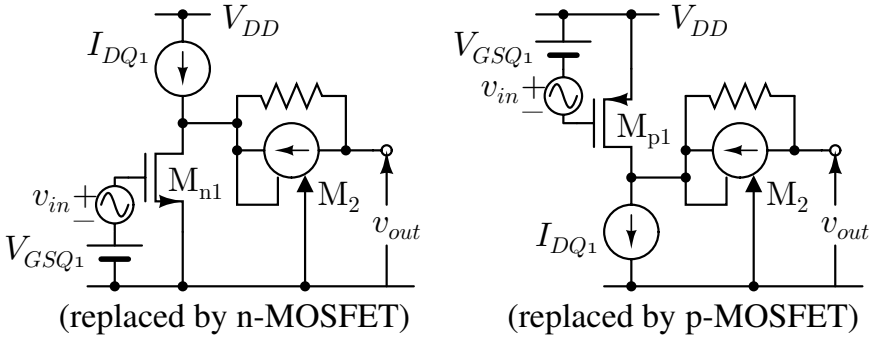
and I_{DQ} is calculated from Eqs. (3.7) or (3.13). After replacing simplified elements with MOSFETs, we must add DC current and voltage sources to adjust the discrepancies in DC currents and voltages. For example, consider the circuit shown in Fig. 3.12(a) which is obtained by the replacement. From the calculation, V_{GSQ1} and V_{GSQ2} are obtained, however we can not necessarily guarantee $V_{GSQ1} + V_{GSQ2} = V_{DD}$. Thus we need a DC voltage source V_{shift} between the two gates to have $V_{GSQ1} + V_{GSQ2} + V_{shift} = V_{DD}$ as shown in Fig. 3.12(b). Similarly, if $I_{DQ1} \neq I_{DQ2}$ in Fig. 3.12(a), we must add a current source I_{shift} as shown Fig. 3.12(b), where $I_{shift} = I_{DQ1} - I_{DQ2}$. All these DC voltage and current sources, V_{shift} and I_{shift}, are replaced with well-known DC level shift circuits and DC current source ones.

Minimization of the number of DC voltage sources

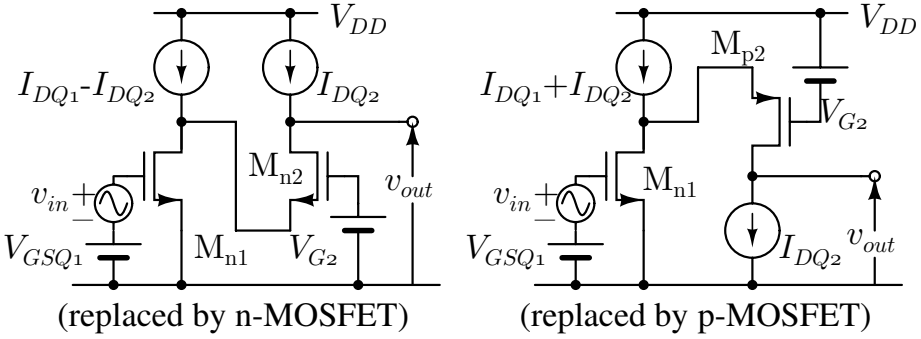
The number of DC voltage sources varies according to the order in which node voltages are calculated. For example, consider the circuit shown in Fig. 3.13(a). If



(a) Circuit with simplified elements



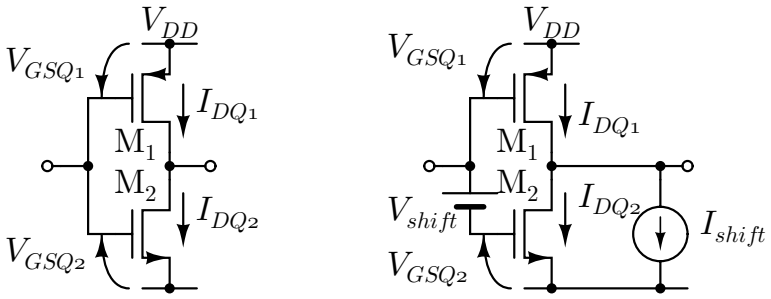
(b) Replacement of the first element M_1



(c) Replacement of the second element M_2

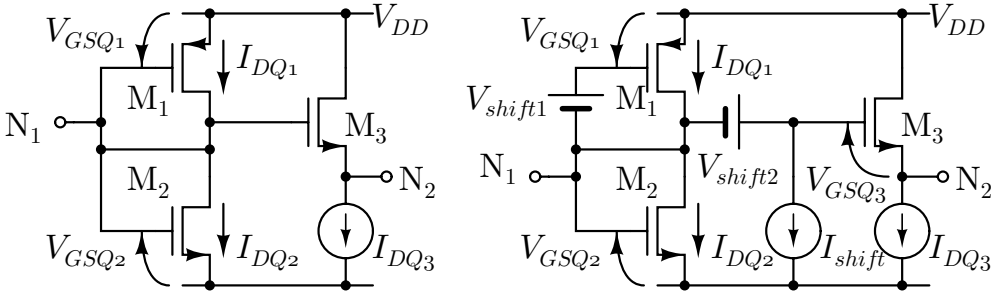
Figure 3.11: Replacement of simplified elements by MOSFETs.

voltage of the node N_1, V_{N1} is given by V_{GSQ2}, two DC voltage sources are needed as shown in Fig. 3.13(b) where V_{shift1} is needed to have $V_{GSQ1} + V_{GSQ2} + V_{shift1} = V_{DD}$ and V_{shift2} sets voltage of the node N_2 high enough to keep the MOSFET of the current source I_{DQ3} operating in saturation region. However, if V_{N1} is given by $V_{DD} - V_{GSQ1}$, only one DC voltage source is necessary as shown in Fig. 3.13(c) when V_{DD} is high enough. Thus, the order in which node voltages are calculated is

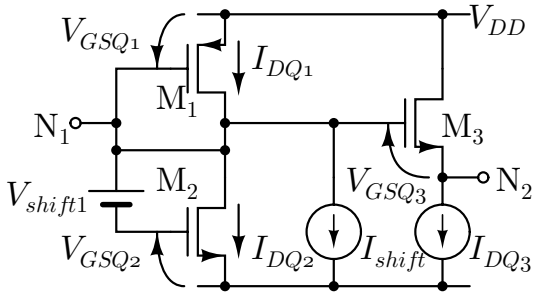


(a) Conflict of V_{gs} (b) Addition of V_{shift} and I_{shift}

Figure 3.12: DC voltage and current adjustments.



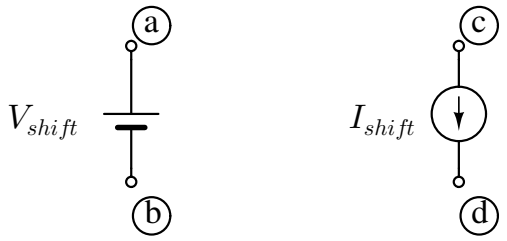
(a) Conflict of V_{gs} (b) When V_{N1} is given by V_{GSQ2}



(c) When V_{N1} is given by $V_{DD} - V_{GSQ1}$

Figure 3.13: Variance of the number of DC voltage sources.

permuted to minimize the number of DC voltage sources.



(a) Voltage source. (b) Current source.

Figure 3.14: Floating sources.

Realization of Floating DC Voltage and Current Sources

In order to properly bias transistors, we often need floating DC voltage sources and DC current ones as shown in Fig. 3.14. The DC voltage source of Fig. 3.14(a) can be realized by Fig. 3.15(a) and (b). Fig. 3.15(a) gives a wide range of voltage of

$$V_{shift} = R_{shift} I_0 \tag{3.27}$$

by selecting R_{shift} and I_0. On the other hand, Fig. 3.15(b) only gives a voltage of

$$V_{shift} = V_{GS} \tag{3.28}$$

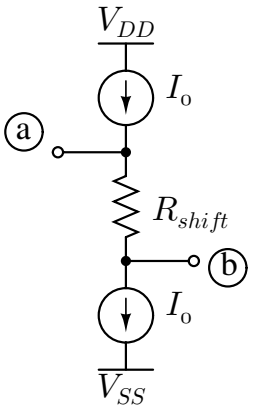
which has a quite narrow range determined by the size and current of the MOSFET. The current of the two current sources of Fig. 3.15(a) must be matched and its internal impedance of I_s must be high enough to obtain high performance of the level shifts. When the DC voltage source is connected between a drain terminal and source terminal as shown in Fig. 3.16(a), its role is to keep M_1 operating in the saturation region and transfer the current signal from M_1 to M_2. Therefore, the DC voltage source in Fig. 3.16(a) can be realized by two current mirror circuits as shown in Fig. 3.16(b).

The floating current source can be realized by using two current sources which are connected to the DC sources as shown in Fig. 3.17.

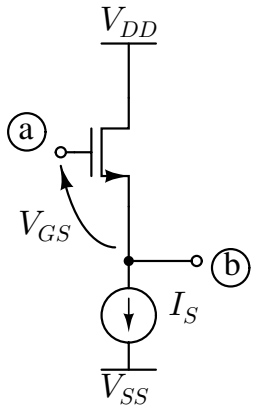
All the current sources of Figs. 3.15 and 3.17 are realized by n-MOSFET or p-MOSFET. Errors caused by mismatch of the current sources can be adjusted by the optimization process explained in the next section.

3.4.3 Optimization of MOSFET Circuits

Now, we obtained a MOSFET circuit which has been created by using simplified elements. In the synthesis, we assumed that the parameters, λ, K (only for the case

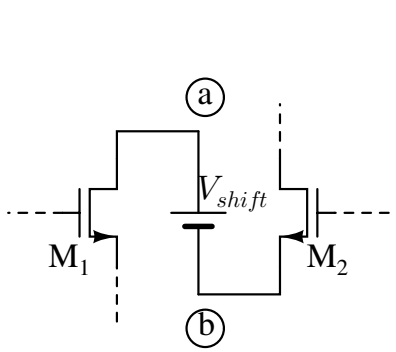


(a) Level shift by a resister.

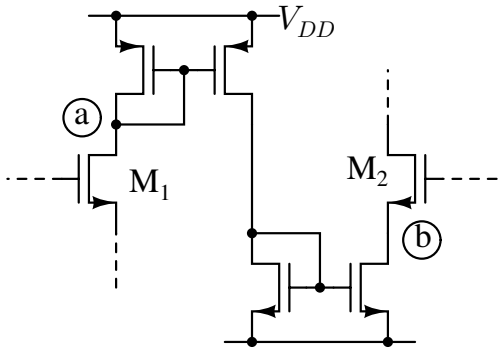


(b) Level shift by a MOSFET.

Figure 3.15: Floating voltage sources.



(a) Level shift by a DC voltage source.



(b) Level shift by two current mirror circuits.

Figure 3.16: Floating voltage sources between drain and source terminals.

of nonlinear circuits), and r_d were constant. However, MOSFETs of the created circuits must have different values of the parameters and thus there must be errors in circuit characteristics. Therefore we must optimize the circuits to meet the given specifications. For this optimization, we can use any optimizers available in the market, since these optimizers are effective if the circuit structures are given.

The whole synthesis flow of the proposed method is summarized in Fig. 3.18.

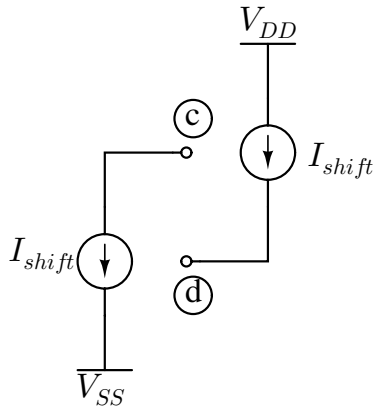


Figure 3.17: Floating current source.

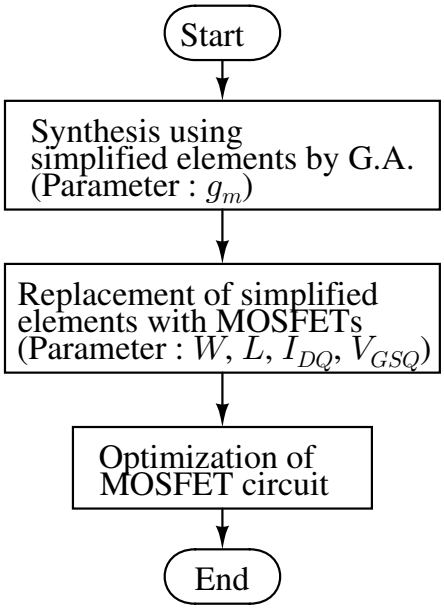


Figure 3.18: Synthesis flow.

3.5 Design Examples

To demonstrate the capability of the proposed method, two automated design examples, a trans-impedance amplifier and a cubing circuit are shown. Also, to show the superiority of the proposed method, benchmark tests are done. Then, the design of a differential voltage amplifier with a specified input referred noise is shown as more complicated specifications. 0.35μm CMOS process parameters are used to replace the simplified elements by MOSFETs and perform simulations of MOSFET circuits.

3.5.1 Trans-impedance Amplifier

Specifications

The specifications of a trans-impedance amplifier are shown in Table 3.2. Among these specifications, we selected Gain and Input Resistance as small signal specifications to create a circuit with simplified elements.

The fitness function f_i is defined as

$$f_i = \min\left(1, 1 + \tanh\left(\frac{x_{result} - x_{spec}}{x_{th}}\right)\right) \tag{3.29}$$

for Gain and

$$f_i = \min\left(1, 1 - \tanh\left(\frac{x_{result} - x_{spec}}{x_{th}}\right)\right) \tag{3.30}$$

for Input Resistance where x_{result} is the simulation result, x_{spec} is the specification, and x_{th} is a constant which determines the weighting value of the function. Here, $\tanh(x)$ and the differentiation of $\tanh(x)$ are given as

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \tag{3.31}$$

$$(\tanh(x))' = \frac{4}{(e^x + e^{-x})^2}, \tag{3.32}$$

$$(\tanh(x))'' = \frac{-4(e^x - e^{-x})}{(e^x + e^{-x})^3}. \tag{3.33}$$

By these equations, we can see $\tanh(x)$ has a maximum differential coefficient at $x = 0$. Thus we can selectively evaluate errors of the characteristics around the specifications by using $\tanh(x)$ as a fitness function.

Synthesis Results

Figs. 3.19(a), (b) and (c) are the best circuits obtained at the generation 0, 1 and 2, respectively. Fig. 3.19(c) satisfies specifications of Gain and Input Resistance. Replacing the simplified elements of Fig. 3.19(c) by MOSFETs, we obtain the circuit shown in Fig. 3.19(d). In the circuit, the MOSFETs, $M_5 \sim M_{10}$ are added to bias the MOSFETs. Performances of the generated circuit meet all the specifications as shown in Table 3.2 and no optimization of the circuit was necessary for this case.

3.5.2 Cubing Circuit

Specifications

We consider a cubing circuit whose output current I_{OUT} is given by

$$I_{OUT} = H \cdot I_{IN}(I_{IN} - I_0)(I_{IN} + I_0) \tag{3.34}$$

Table 3.2: Design specifications and synthesis results of a trans-impedance amplifier.

	Specifications	Results
Gain [dBΩ]	> 100	109
Input Resistance [Ω]	< 100	92.5
Power [mW]	< 0.6	0.34
Area [μm^2]	< 400	256
Supply voltage [V]	3.0	3.0

where I_{IN} is the input current, I_0 is a constant current which determines the cubic characteristics, and H is a constant having a dimension of [A^{-2}] which determines the magnitude of the output current.

Here, we select,

$$I_0 = 35[\mu\mathrm{A}], \tag{3.35}$$

$$H = 5 \times 10^8[\mathrm{A}^{-2}], \tag{3.36}$$

and the input current range is

$$-50[\mu\mathrm{A}] \leq I_{IN} \leq 50[\mu\mathrm{A}]. \tag{3.37}$$

The nonlinear parameters are set as

$$\alpha = \alpha_n = \alpha_p = 700[\mu\mathrm{A/V}^2] \tag{3.38}$$

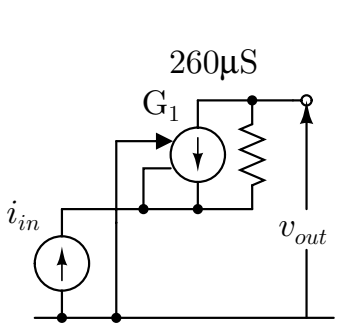
calculated from $W_n/L_n = 10.0\mu$m$/1.0\mu$m and $W_p/L_p = 25.0\mu$m$/1.0\mu$m that are selected for this example. If we can not obtain any satisfactory circuit, these parameters must be amended appropriately.

The output currents given by Eq. (3.34) must be within a given margin of error, thus we utilize here a convex function of error as a fitness function given by,

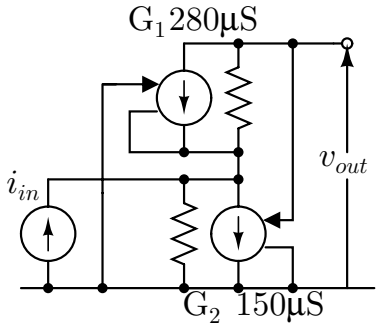
$$f_i = \frac{1}{m}\sum_{n=1}^{m}\left(\min\left(1, \frac{2}{\left|\frac{error(n)}{margin}\right|^2 + 1}\right)\right), \tag{3.39}$$

$$error(n) = \frac{I_{OUT}(n) - I_{IDEAL}(n)}{I_{MAX}} \times 100, \tag{3.40}$$

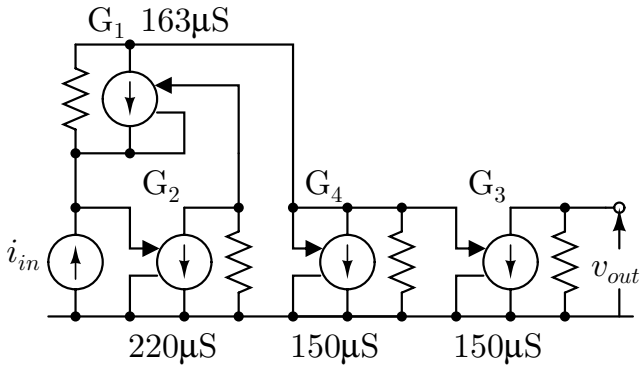
where m is the number of sampling points, $I_{IDEAL}(n)$ and $I_{OUT}(n)$ are the theoretical and simulated output currents at the sampling point n, I_{MAX} is the maximum output



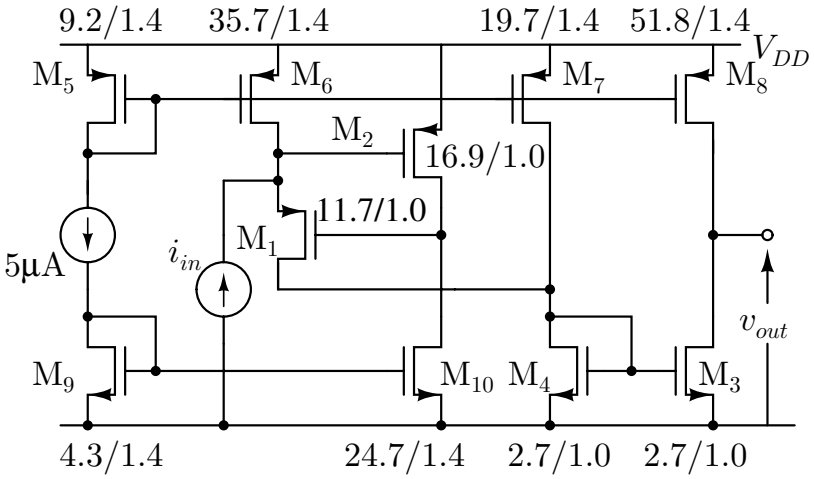
(a) Best circuit at generation 0



(b) Best circuit at generation 1



(c) Best circuit at generation 2



(d) Best circuit at generation 2 replaced by MOSFETs

Figure 3.19: Synthesis result of trans-impedance amplifier.

current, and $margin$ is an acceptable error from I_{IDEAL}, respectively. We select $margin = 3\%$. To emphasize the error, we employ a square function of $error(n)$ in Eq. (3.39). Since the differentiation of $\dfrac{1}{x^2+1}$ are given as

$$\left(\frac{1}{x^2+1}\right)' = -\frac{4x}{(x^2+1)^2}, \tag{3.41}$$

$$\left(\frac{1}{x^2+1}\right)'' = -\frac{4(x+1)(x-1)(x^2+1)}{(x^2+1)^4}, \tag{3.42}$$

$\dfrac{1}{x^2+1}$ has a maximum differential coefficient at $x = \pm 1$, i.e., $error(n) = \pm margin$. Thus, as well as $\tanh(x)$, we can selectively evaluate errors of the characteristics around the specifications by using $\dfrac{1}{x^2+1}$ as a fitness function .

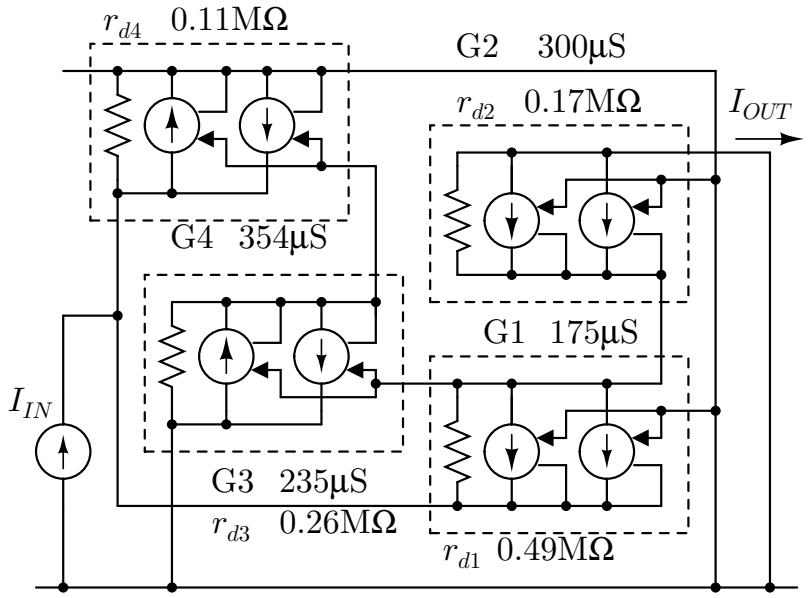
Synthesis Results

The best circuits at the generation 0, 9, 21 and 32 and their output currents are shown in Fig. 3.20(a)∼(d) and Fig. 3.21, respectively. The output current of the circuit Fig. 3.20(d) almost coincides with the ideal cubic curve.

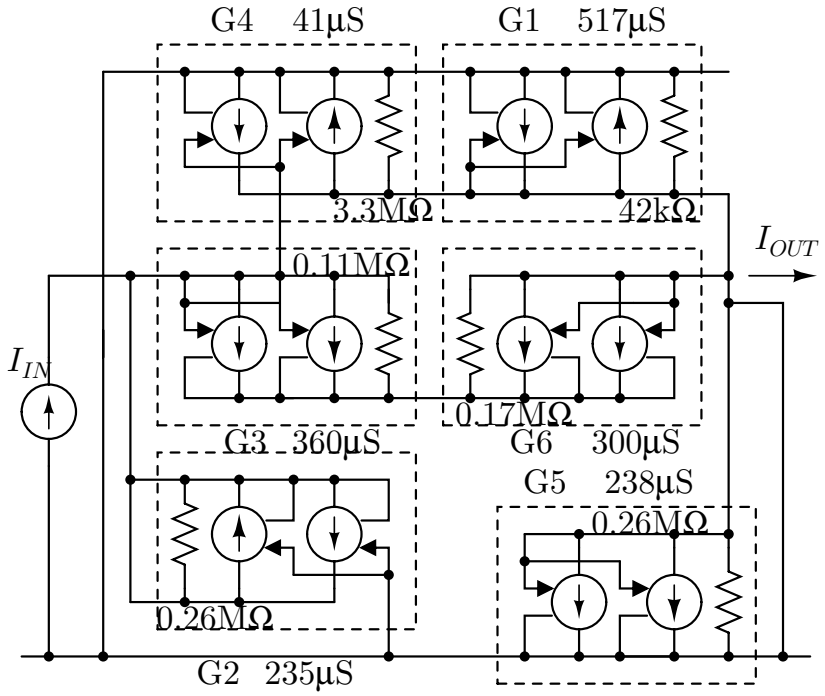
By replacing the simplified elements of the best circuit of the generation 32 with MOSFETs, we obtain the circuit given in Fig. 3.22 where $M_{10} \sim M_{15}$ give the DC bias currents and $V_1 \sim V_4$ are DC voltage sources to adjust the DC bias voltage. For the simplicity of the circuit diagram, Fig. 3.22 still includes DC voltage and current sources which are replaced with MOSFET circuits during the simulation. The output current of Fig. 3.22 is plotted in Fig. 3.23(a) and errors from the ideal currents are shown in Fig. 3.23(b). The error current after the replacement of simplified elements with MOSFETs becomes large. This error is resulted from the difference between the fixed parameter values of $K(= 2\alpha)$, r_d, and λ and the actual ones and also from the non-ideality of level shift circuits. Thus we need to optimize the parameters of Fig. 3.22 using an appropriate optimizer available in the market. Here The optimizer served in HSPICE was used. The values of the parameters before and after the optimization are given in Fig. 3.22, where the sizes of MOSFETs before the optimization are given in the parenthesis. The output current of the final circuit is given by the solid line (c) of Fig. 3.23(a) which shows quite good agreements with the ideal characteristics (d).

3.5.3 Benchmark Tests

In order to confirm the abilities of the proposed method, we compare the method with conventional ones which directly create analog circuits. The most significant



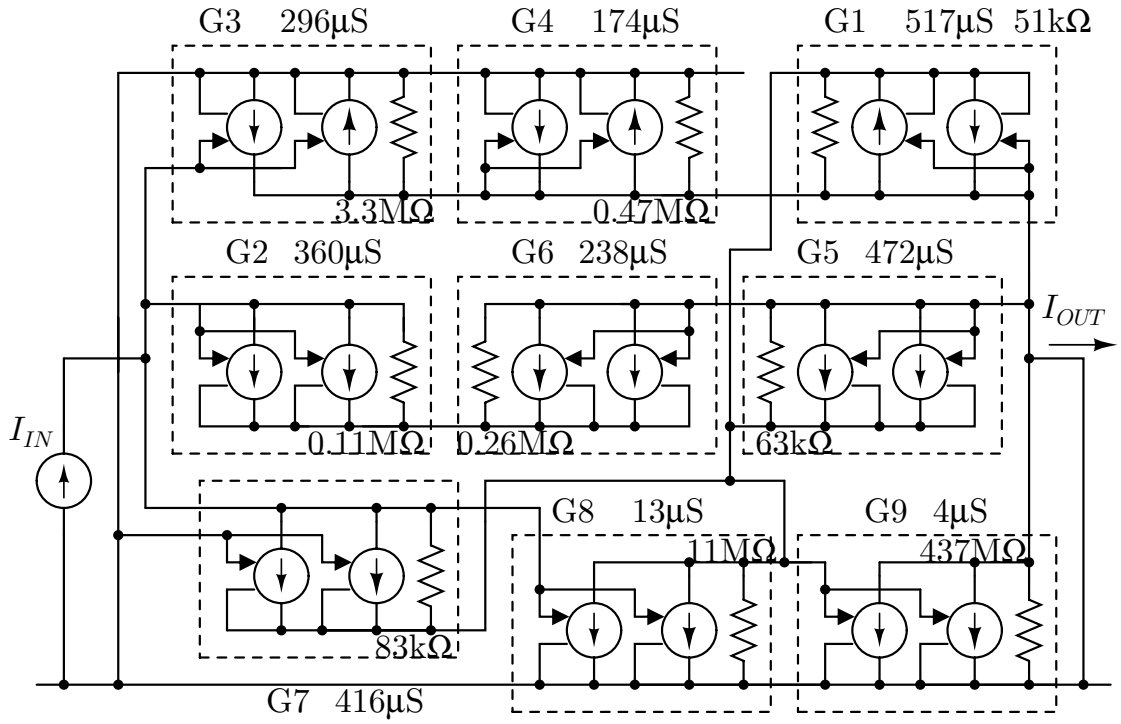
(a) generation 0



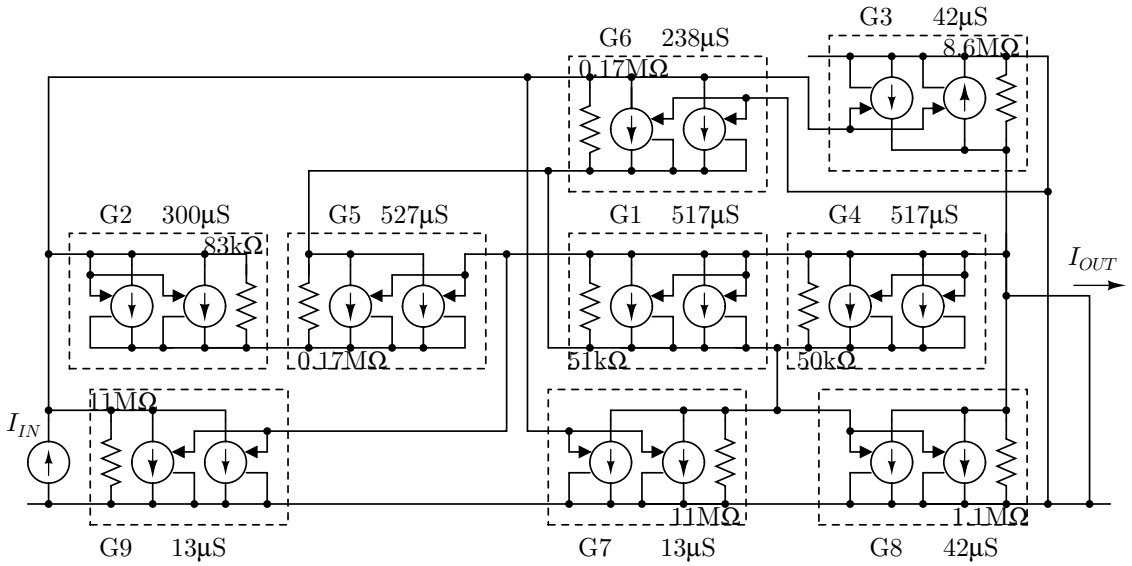
(b) generation 9

Figure 3.20: Best circuits of generation 0, 9, 21, and 32.

feature of the proposed method is to use simplified elements at the first stage of



(c) generation 21



(d) generation 32

Figure 3.20: (Continued)

circuit creation. Thus, we must compare two synthesis processes, one begins from simplified elements and the other directly creates a circuit, by using the same auto-

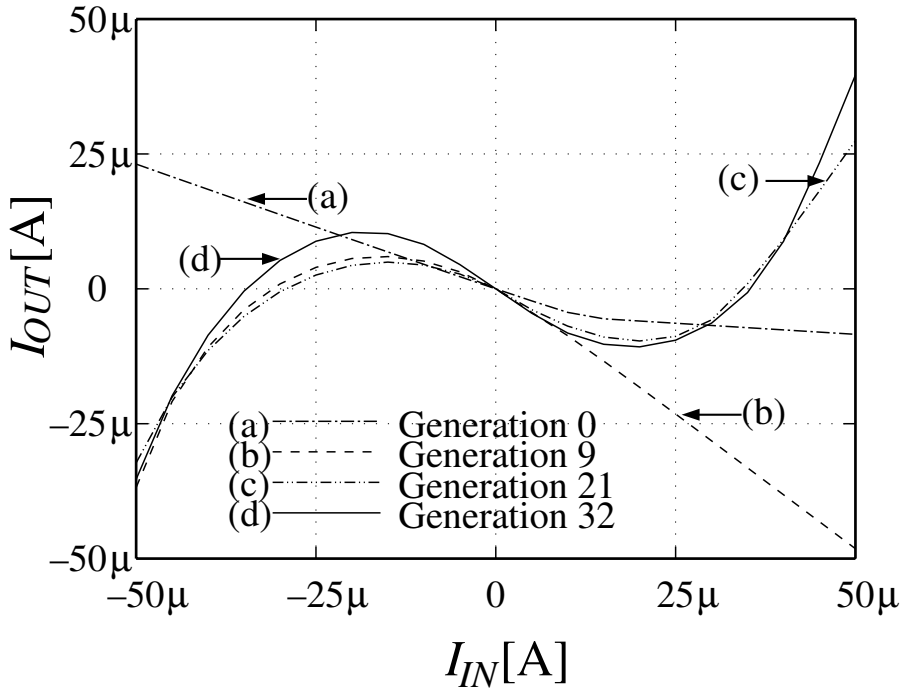


Figure 3.21: DC characteristics of Fig. 3.20.

mated synthesis program.

Since it is quite difficult to find a CAD program whose algorithm is clearly opened and amendable by the users, the program of the reference [34] that was coauthored by one of the authors of a publication related to this dissertation is used. As the targets of the benchmark test, we selected two circuits, a squaring and a cubing circuits. Both circuits were synthesized with a cell-based structure by the program of the reference [34].

The results of the benchmark tests are shown in Table 3.3, where 'Proposed' means that the circuits were obtained using simplified elements and 'Conventional' means that MOSFET circuits were directly created. In the test, we performed trials to obtain five successful circuits for each case. As the measures of evaluation, we employed the average number of generations and the average simulation time required to get a successful circuit, and the success rate which is an average rate of the number of successful circuits and trials. From the results, we see the proposed method can generate a squaring circuit about 10 times faster than the conventional one. In the case of the cubing circuit, the synthesis time of the proposed method is only 2.5 times shorter than that of the conventional one, but the success rate is five times higher than that of the conventional one. From these benchmark tests, we can say the proposed method can generate a circuit faster at a higher success

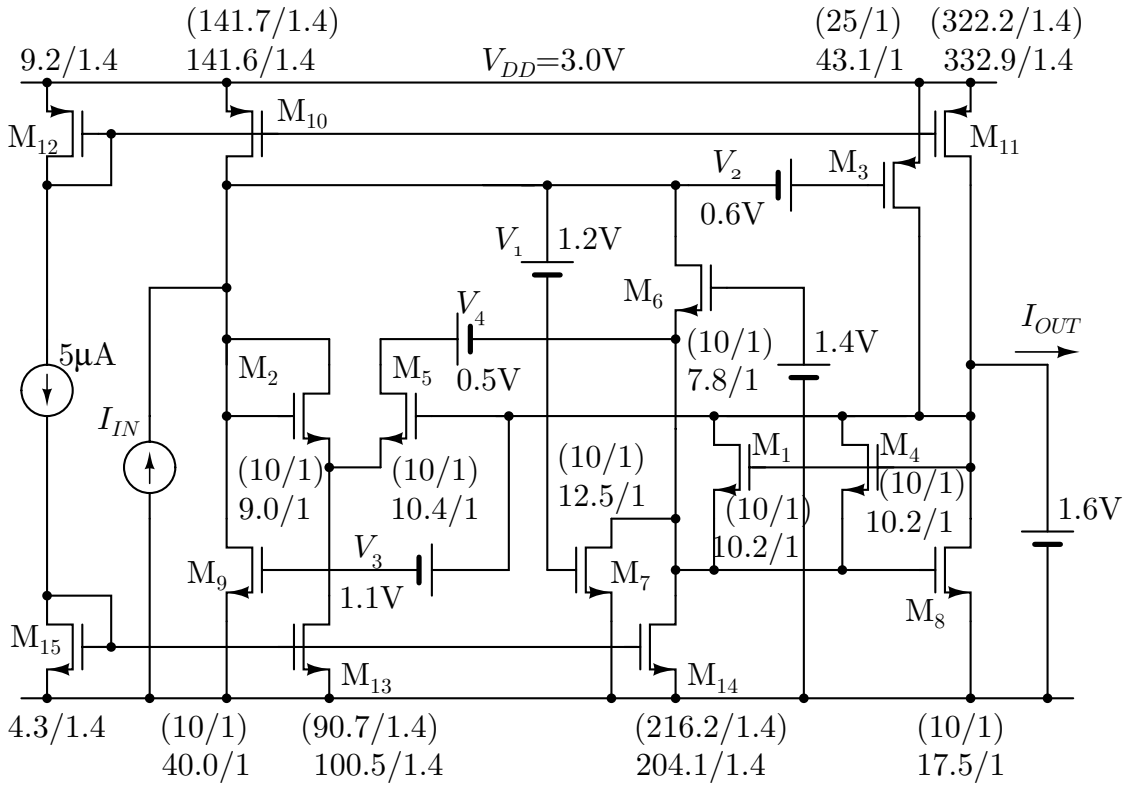


Figure 3.22: Circuit of Fig. 3.20(d) replaced by MOSFETs.

Table 3.3: Results of benchmark tests.

		Average number of generations	Average synthesis time [sec.]	Success rate
Squaring	Conventional	323.2	4454.6	63%
	Proposed	39.3	391.1	83%
Cubing	Conventional	328.8	4539.3	9%
	Proposed	177.0	1797.3	50%

rate than conventional one. The proposed method can be universally applied to any conventional automated analog circuit design program.

3.5.4 More complicated specifications

Since we can reduce the design time, we can add more complicated specifications such as noise characteristics which human can hardly treat. Here, we consider the design of a differential voltage amplifier with a specified input referred noise. The

Table 3.4: Design specifications of a differential voltage amplifier.

	Specifications
(1) $r_{in1}[\Omega]$	$\geq 10^{20}$
(2) $r_{in2}[\Omega]$	$\geq 10^{20}$
(3) $(\phi_1 - \phi_2)_{dif}$ [deg.]	within 180 ± 0.5
(4) A_{dif1} [dB]	≥ 100
(5) A_{dif2} [dB]	within $A_{dif1} \pm 1$
(6) $(\phi_1 - \phi_2)_{com}$ [deg.]	within 0 ± 0.5
(7) A_{com1} [dB]	≤ 0
(8) A_{com2} [dB]	within $A_{com1} \pm 1$
(9) Bias deviation ΔV[V]	≤ 0.05
(10) V_{inoise} [nV/$\sqrt{\text{Hz}}$]	≤ 200 at 1kHz
(11) Power [μW]	≤ 500
(12) Area [μm^2]	≤ 500
Supply voltage[V]	3.0

specifications of the amplifier are shown in Table 3.4. The details of the specifications are explained in the next Chapter, however, we can see that more than double of specifications must be considered compared to the design of single-phase circuits.

Five trials of the differential voltage amplifier (Amplifier) were performed, and the average number of generations and average simulation time to get a successful circuit, and the success rate are shown in Table 3.5, with the results of the design of trans-impedance amplifier (TIA) explained in Section 3.5.1 for comparison. These results show the synthesis still becomes time-consuming and the success rate decreases when the number of specifications increases.

3.6 Conclusions

This chapter proposed an automated design of analog circuits which can extremely reduce the design time. Instead of using complicated transistor models, simplified models of transistor is introduced at the first stage of the topology synthesis. After a circuit topology is obtained using the simplified elements, the simplified elements are replaced by transistors and the transistor parameters are optimized to meet the

Table 3.5: Results of five trials of trans-impedance amplifier and differential voltage amplifier.

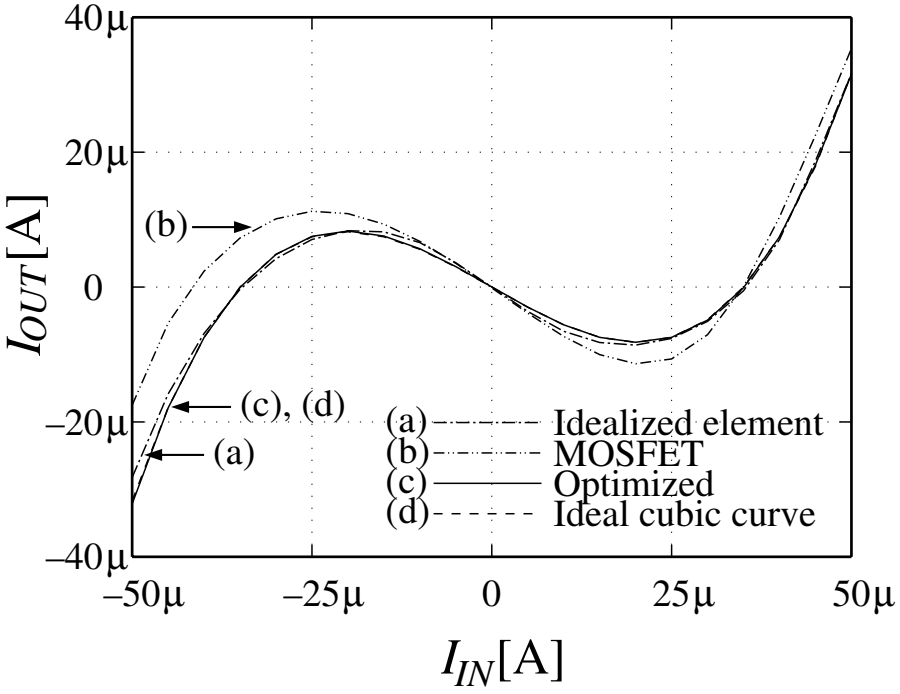
	Average number of generations	Average synthesis time [sec.]	Success rate
TIA	3	30	100%
Amplifier	37	4411	80%

given specifications.

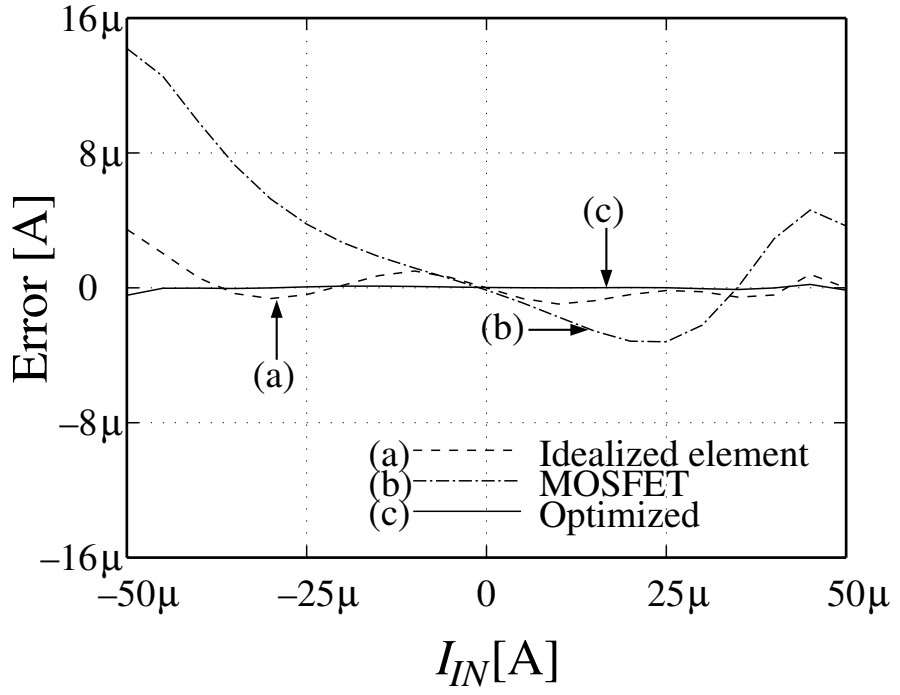
The obtained circuits only contain transistors that operate in active region and consequently the circuits become quite reasonable. Also, the proposed method is able to generate a reasonable circuit flexibly according to the given specifications. The synthesis results show the proposed method is useful for the design of both of linear and nonlinear circuits and the benchmark tests of design examples show that the proposed CAD is more than 10 times faster than the previous one. The proposed method can be universally applied to any conventional automated analog circuit design program.

Although this chapter concentrates the discussion on CMOS analog circuits, the proposed method can be applied to bipolar transistor circuits by using simplified elements that express the basic characteristics of bipolar transistor.

However, it must be noted that the synthesis still tends to be time-consuming when specifications become complicated as in the design of differential circuits. A solution to reduce the design time further is described in the next Chapter.



(a) DC characteristics



(b) Error from the ideal curve

Figure 3.23: DC characteristics of the obtained cubing circuit.

Chapter 4

Automated Design of Analog Circuits Accelerated by Reuse of Genetic Operations

4.1 Introduction

In Chapter 3, in order to effectively create a new circuit topology without giving any initial circuit, a simplified model of MOSFET that was represented by a simple voltage controlled current source with only one variable parameter to be adjusted was introduced. This can shrink the search space and consequently reduce the design time extremely and obtain quite reasonable circuits.

However, the synthesis still tends to be time-consuming when the number of specifications increases. For example, the synthesis of fully differential circuits, in which both of differential mode and common-mode specifications must be considered, is still quite difficult and time-consuming as described in Section 3.5.4. This seems to be because the genetic operations are executed randomly and effective genetic operations which improve circuit performances decrease as the number of specifications increases.

This chapter proposes reuse of genetic operations to speed up the synthesis. Analog circuit engineers often design a circuit by revising well-designed circuits or using a part of the well-designed circuits, and some programs follow this manner[37], [39]. On the other hand, the proposed architecture does not reuse circuit topologies but genetic operations that can improve a specified circuit performance. This must be more flexible. The reuse of genetic algorithm together with a simplified MOSFET model, we can extremely reduce the design time and improve the success rate to 100% to obtain a satisfactory circuit. The effectiveness of this method is demon-

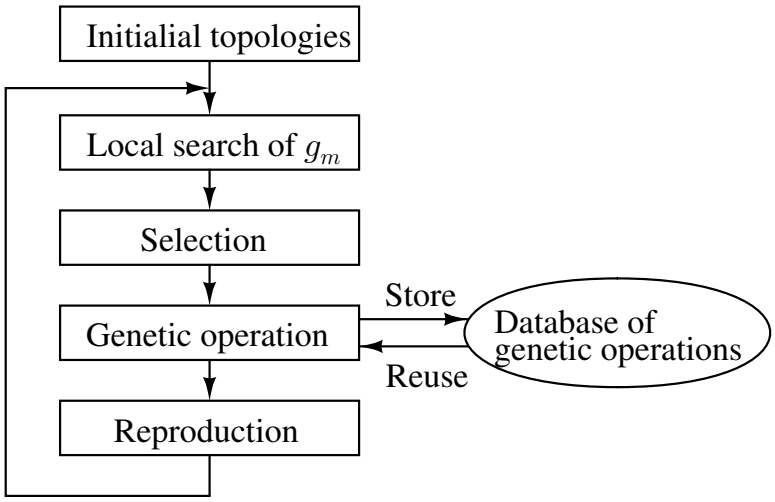


Figure 4.1: Genetic algorithm with reuse of genetic operations

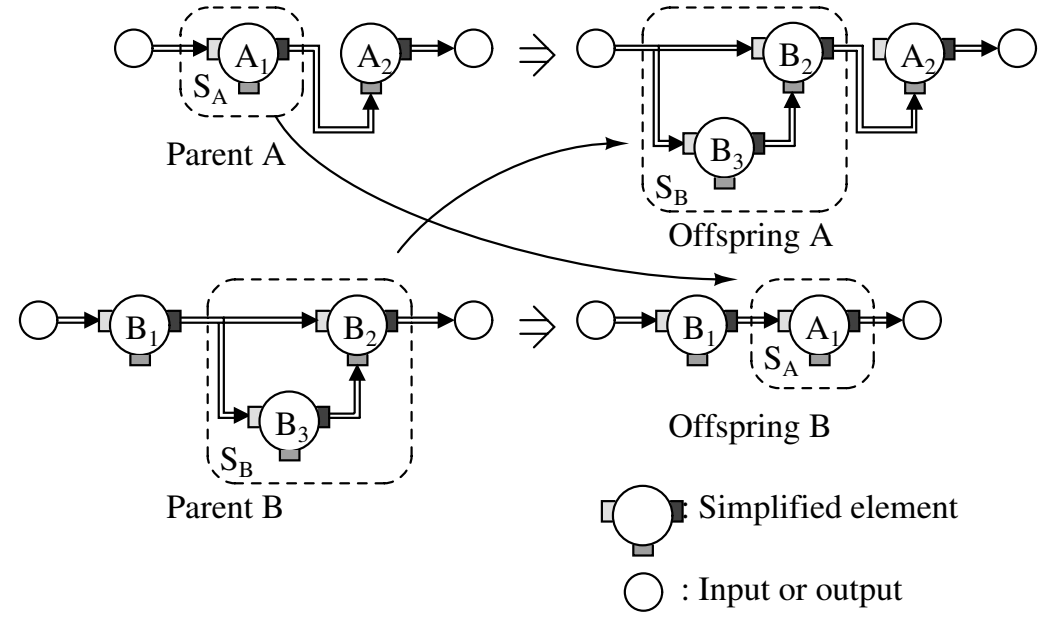
strated through three design examples which are quite difficult to design even by skilled analog circuit engineers.

4.2 Genetic Algorithm with Reuse of Stored Operations

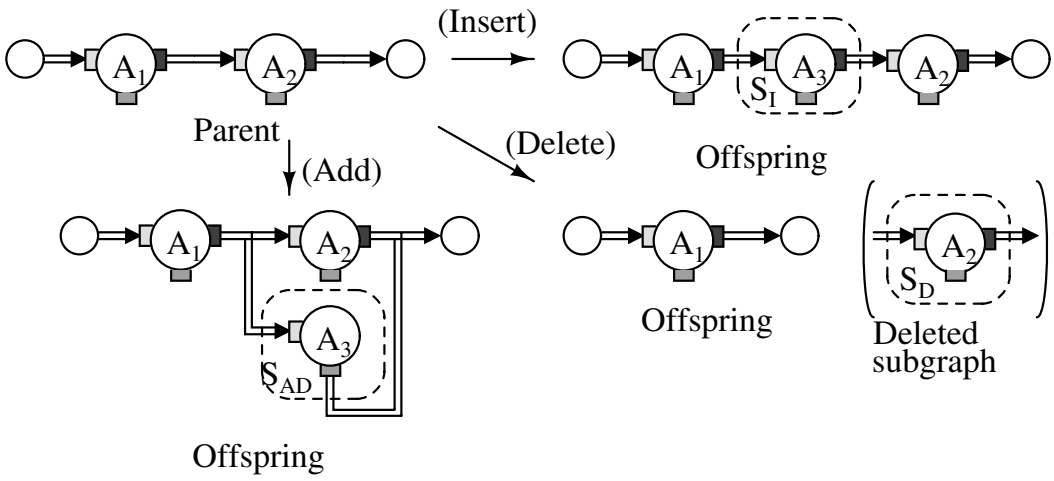
4.2.1 Genetic Operation

In the previous chapter, a genetic algorithm was employed to create circuits using simplified elements. In order to speed up the synthesis, reuse of the genetic operations which improve the circuit performances is introduced in this chapter. The procedure of the genetic algorithm is basically the same as explained 3.3. The four phases, i.e. local search of g_m, selection, genetic operation, and reproduction are iterated as shown in Fig. 4.1 until the given specifications are satisfied.

At the genetic operation stage, two types of genetic operation, Crossover and Mutation are used to reproduce new genomes in the next generation. Here again, Crossover and Mutation are explained using Fig. 4.2 where circuit graphs of differential circuits are depicted. Crossover exchanges subgraphs between the two selected genomes as shown in Fig. 4.2(a) where the two new genomes, Offspring A and B, are reproduced from the two Parents A and B by exchanging the subgraphs S_A and S_B encircled by the dotted lines. Mutation inserts, adds, or deletes a subgraph as shown in Fig. 4.2(b). Then, the operations which improve circuit performances are stored into the Database and reused to obtain better circuits.



(a) Crossover



(b) Mutation

Figure 4.2: Example of stored genetic operations

4.2.2 Store of Genetic Operations

The database of genetic operations is built up at each generation as follows. Here we assume there are N specifications, $Spec_1, Spec_2, \cdots$, and $Spec_N$ and the fitness of the i-th specification at the j-th generation is designated as f_i^j. At the local search stage, f_i^j is compared with f_i^{j-1}. If $f_i^j > f_i^{j-1}$ under the condition $f_k^j \geq f_k^{j-1} (k \neq i)$,

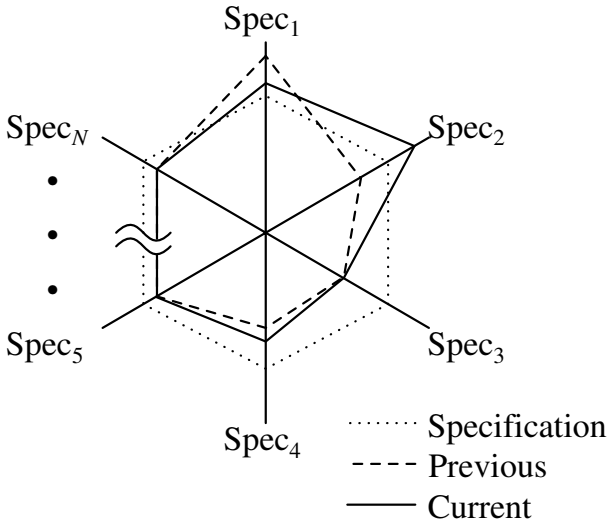


Figure 4.3: Characteristic change when an operation is stored

then the genetic operation which made $f_i^j > f_i^{j-1}$ is stored in the database. That is, the genetic operation which improves the fitness of a specification is stored unless all fitness of all other specifications does not become worse. As an exception, if $f_k^j < f_k^{j-1}$ and the k-th performance still satisfies the k-th specification as shown in Fig. 4.3(Spec$_1$), the operation is stored in the database.

As data of a genetic operation, the following four categories are stored in the database.

(a) type of specifications

(b) improvement ratio of the characteristic

(c) type of genetic operation

(d) topological information

Topological information to be stored is subgraphs where the genetic operation is executed and partial topologies around them. For example, we consider the case that a crossover modifying the parent B to the offspring B as shown in Fig. 4.2(a) improves the fitness. The exchanged subgraphs S_A and S_B and a partial topology of Parent B to which S_B is connected are stored.

4.2.3 Reuse of Genetic Operations

At the genetic operation stage of Fig. 4.1, the stored operations are reused based on the specifications which must be improved. Suppose that the gain of an amplifier,

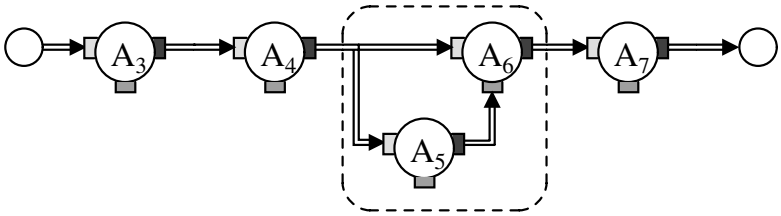


Figure 4.4: A genome for which the operation of Fig. 4.2 is reused

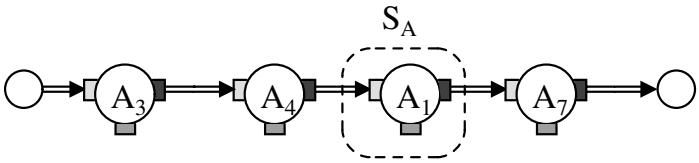


Figure 4.5: New topology obtained by reuse of crossover

a part of which is given by a topology shown in Fig. 4.4 for example, is to be improved. The database is searched within the group of gain enhancing operation to find a similar topology with that of Fig. 4.4. The topology consisting of A_5 and A_6 circled by a dotted line of Fig. 4.4 is quite similar to the part S_B of the Parent B in Fig. 4.2(a). The Parent B is revised by the crossover and S_B is exchange with S_A to get an Offspring B. If this crossover is stored to improve the gain, the operation is adopted to A_5 and A_6 to obtain a new topology of Fig. 4.5. If more than two operations are found to improve the gain, one which has the highest improvement ratio is selected. If no candidates are found, general genetic operations without reuse of stored data is executed using current population.

4.2.4 Fitness and Fitness Function

As explained in Section 3.3.3, we evaluate circuits by fitness functions at the selection stage. The fitness function of each specification can be defined by users and we employed the fitness functions explained in Section 3.3.3 also in this chapter. All the fitness functions f_i are normalized to have a value between 0 and 1 as,

$$0 \leq f_i \leq 1, \tag{4.1}$$

where f_i becomes 1 when a specification is satisfied. The fitness F to evaluate the total performance of a circuit is defined by

$$F = \sum_{i=1}^{N} f_i \tag{4.2}$$

where N is the number of specifications. If all the given specifications are satisfied, then $F = N$.

4.3 Synthesis Procedure

The synthesis procedure is the same as explained in the previous Chapter. Here we summarize the procedure. At the first stage of the synthesis, circuits are generated using simplified elements by the genetic algorithm with reuse of genetic operations. Then, all the simplified elements in a circuit obtained by the synthesis are replaced by MOSFETs and the created MOSFET circuit is optimized to meet the given specifications if necessary.

4.4 Design Examples

To demonstrate the capability of the proposed method, we show one linear circuit and two nonlinear circuit examples. The linear circuit is a differential voltage amplifier with a specified input referred noise which is taken up in Section 3.5.4. One of the nonlinear circuits is a differential circuit (Cubic-square circuit) whose differential mode output current is proportional to a cubic function of the differential input voltage and common mode output current is proportional to a square function of the common mode input voltage. The other is a dB-Linear VGA, which is an amplifier having a voltage controllable gain. The gain given in dB is proportional to a controlled DC voltage. These three examples are quite difficult to design even by skilled analog circuit engineers. Also, to show the superiority of reuse of genetic operations, we compared the synthesis with and without reuse of genetic operations. 0.35μm CMOS process parameters were used to replace the simplified elements by MOSFETs and perform simulations of MOSFET circuits.

4.4.1 Differential Voltage Amplifier With a Specified Input Referred Noise

Specifications

The specifications of the differential amplifier are shown in Table 4.1 again. r_{in1} and r_{in2} are input resistance at the input terminal 1 and 2, respectively. For the symmetry of characteristics, gain has to be evaluated separately at the two outputs. Therefore, as specifications of the gain, we define the gains, A_{dif1} and A_{dif2} for

differential mode as

$$A_{dif1} = \frac{v_{out1}}{v_{in1} - v_{in2}} \tag{4.3}$$

$$A_{dif2} = \frac{v_{out2}}{v_{in1} - v_{in2}} \tag{4.4}$$

and for common-mode, A_{com1} and A_{com2} as

$$A_{com1} = \frac{v_{out1}}{v_{in1} + v_{in2}} \tag{4.5}$$

$$A_{com2} = \frac{v_{out2}}{v_{in1} + v_{in2}}. \tag{4.6}$$

$(\phi_1 - \phi_2)_{dif}$ and $(\phi_1 - \phi_2)_{com}$ are phase difference between the differential and common mode output signals, respectively.

Circuits obtained by using simplified model satisfy only AC specifications. In order to maintain the DC bias stability, a common mode feedback is generally required. Without common mode feedback, the DC biasing point at high impedance nodes becomes stuck to the ground or the power supply voltage due to the mismatch of DC currents at the nodes. To make the node voltage stable at high impedance nodes, we specify the Bias deviation of Table 4.1 which is defined by the DC voltage change when a current of $\Delta I = 5\mu$A is supplied to the node. In the table, $\Delta V < 0.05$V is given. This means that the DC resistance R at the node is,

$$R < \frac{\Delta V}{\Delta I} = 10\text{k}\Omega. \tag{4.7}$$

This value is much less than the drain resistance $r_d (\fallingdotseq 300\text{k}\Omega)$. By this specification, an appropriate common mode feedback circuit can be installed. V_{inoise} in Table 4.1 is an input-referred noise voltage at 1kHz.

We use the fitness function f_i given as

$$f_i = \min\left(1, 1 + \tanh\left(\frac{x_{result} - x_{spec}}{x_{th}}\right)\right) \tag{4.8}$$

for the specifications (1), (2), (4) and

$$f_i = \min\left(1, 1 - \tanh\left(\frac{x_{result} - x_{spec}}{x_{th}}\right)\right) \tag{4.9}$$

for the specifications (7) and (9) where x_{result} is the simulation result, x_{spec} is the specification, and x_{th} is a constant which determines the weighting value of the function. These functions are the same as Eqs. (3.29) and (3.30) used in Section 3.5.1.

Table 4.1: Design specifications and synthesis results of a differential voltage amplifier.

	Specifications	Results
(1) $r_{in1}[\Omega]$	$\geq 10^{20}$	10^{20}
(2) $r_{in2}[\Omega]$	$\geq 10^{20}$	10^{20}
(3) $(\phi_1 - \phi_2)_{dif}$ [deg.]	within 180 ± 0.5	180
(4) A_{dif1} [dB]	≥ 100	110
(5) A_{dif2} [dB]	within $A_{dif1} \pm 1$	110
(6) $(\phi_1 - \phi_2)_{com}$ [deg.]	within 0 ± 0.5	0
(7) A_{com1} [dB]	≤ 0	-17.8
(8) A_{com2} [dB]	within $A_{com1} \pm 1$	-17.8
(9) Bias deviation ΔV[V]	≤ 0.05	0.05
(10) V_{inoise} [nV/$\sqrt{\text{Hz}}$]	≤ 200 at 1kHz	193
(11) Power [μW]	≤ 500	362
(12) Area [μm^2]	≤ 500	393
Supply voltage[V]	3.0	3.0

For the specifications (3), (5), (6) and (8) in which each characteristic is required to be within a given margin, the fitness function can be a convex function of error as

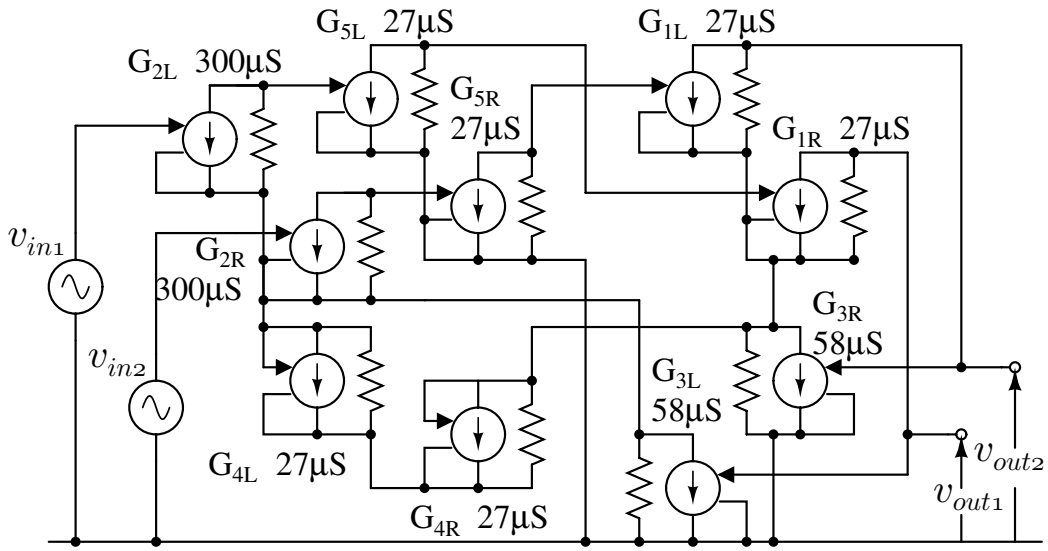
$$f_i = \min\left(1, \frac{2}{\frac{|x_{result} - x_{spec}|}{x_{margin}} + 1}\right), \tag{4.10}$$

where x_{margin} is an acceptable margin from x_{spec}.

If the specifications are satisfied, all these fitness functions have the maximum value of 1 and the fitness F becomes N as explained in Section 4.2.4.

Synthesis Results

Fig. 4.6(a) is the circuit obtained by the synthesis using pairs of simplified elements. G_{iL} and $G_{iR}(i = 1 \sim 5)$ are paired elements. Replacing the simplified elements by MOSFETs, the circuit shown in Fig. 4.6(b) is obtained. In the circuit, the MOSFETs, $M_{6L}, M_{6R} \sim M_{9L}, M_{9R}$ and $M_{10} \sim M_{12}$ are added to appropriately bias the MOSFETs. Performances of the generated circuit meet all the specifications as



(a) Circuit with simplified elements

Figure 4.6: Synthesis result of differential voltage amplifier

shown in Table 4.1. For this linear circuit example, the optimization of the transistor size was not required.

4.4.2 Cubic-Square Circuit

Specifications

We consider a differential circuit whose differential output current is given by

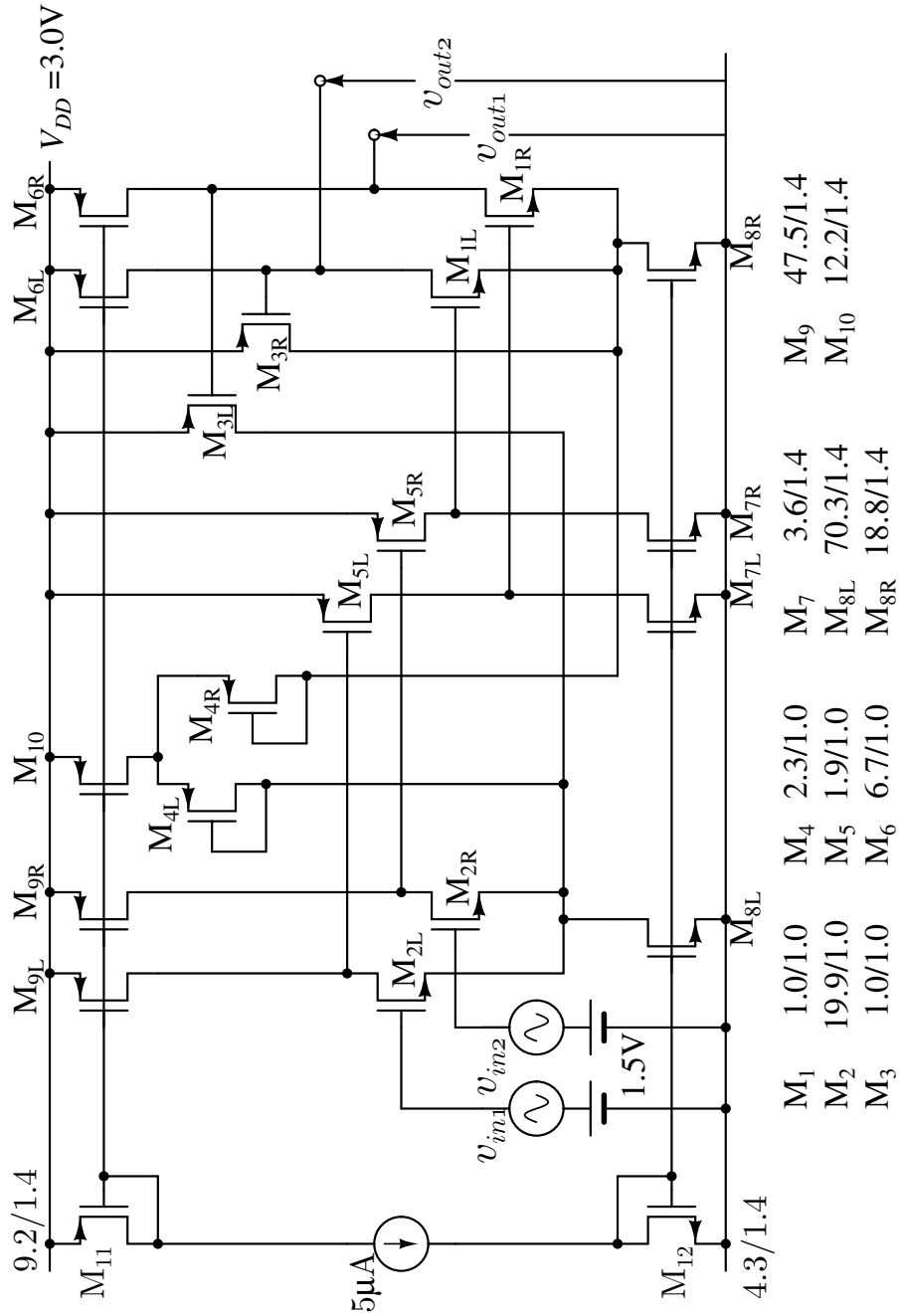
$$I_{OUT1} - I_{OUT2} = G_{DIF}(V_{IN1} - V_{IN2})^3 \tag{4.11}$$

and common mode output current by

$$I_{OUT1} + I_{OUT2} = G_{COM}(V_{IN1} + V_{IN2} - 2V_{INQ})^2 \tag{4.12}$$

where V_{IN1} and V_{IN2} are input voltages, V_{INQ} is a kind of the operating point, and G_{DIF} and G_{COM} are constants which determine the magnitude of the differential and common-mode output currents. we select here $G_{DIF} = 30\mu\text{A/V}^3$ and $G_{COM} = 250\mu\text{A/V}^2$, respectively. The nonlinear parameter α is set as $\alpha = \alpha_n = \alpha_p = 700\mu\text{A/V}^2$ which is the same value given by Eq. (3.38).

From the nature of differential circuits, the following two characteristics are also given as specifications: (a)$I_{OUT1} + I_{OUT2} = 0$ when a differential input is applied and



(b) Circuit replaced by MOSFETs

Figure 4.6: (Continued)

(b)$I_{OUT1} - I_{OUT2} = 0$ when a common-mode input is applied. The bias deviation explained in the previous section is given as a specification, too.

The output currents given by Eqs. (4.11) and (4.12) must be within a given margin of error, thus we utilize here the convex function of error as a fitness function

given by Eq. (3.39), that is,

$$f_i = \frac{1}{m}\sum_{n=1}^{m}\left(\min\left(1, \frac{2}{\left(\frac{error(n)}{margin}\right)^2 + 1}\right)\right), \tag{4.13}$$

$$error(n) = \frac{I_{OUT}(n) - I_{IDEAL}(n)}{I_{MAX}} \times 100, \tag{4.14}$$

where m is the number of sampling points, $I_{IDEAL}(n)$ and $I_{OUT}(n)$ are the theoretical and simulated output currents at the sampling point n, I_{MAX} is the maximum output current, and $margin$ is an acceptable error from I_{IDEAL}, respectively. We select $margin = 3\%$. The total fitness F is calculated by Eq. (4.2) where $N = 5$.

Synthesis Results

Fig. 4.7(a) is the circuit created using pairs of simplified elements. G_{iL} and $G_{iR}(i = 1 \sim 6)$ are paired elements. By replacing the simplified elements of Fig. 4.7(a) and optimizing the size of transistors, we obtain the circuit given in Fig. 4.7(b) where M_7 and M_8 give the DC bias currents and $V_{1L}, V_{1R} \sim V_{4L}$, and V_{4R} are DC voltage sources to adjust the DC bias voltage. Here the optimizer served in HSPICE for the optimization. In Fig. 4.7(b), the sizes of MOSFETs in the parenthesis are the sizes before the optimization. For the simplicity of the circuit diagram, Fig. 4.7(b) still includes DC voltage sources which were replaced with MOSFET circuits during the simulation. The differential and common-mode output currents of Figs. 4.7(a) and (b) are plotted in Figs. 4.8(a) and (b), respectively. The error current just after the replacement of simplified elements with MOSFETs is large, however the output current of the optimized circuit given by the solid line (c) of Fig. 4.8 shows quite good agreements with the ideal characteristics (d).

4.4.3 dB-Linear VGA

Specifications

We consider the design of dB-linear VGA having the following characteristics:

- Variable range of differential gain : -20dB∼20dB
- Range of controlling DC voltage : -0.85V∼1.35V
- Error in gain : ±1dB

Table 4.2: Design specifications and synthesis results of a dB-Linear VGA.

	Specifications	Results
(1) $r_{in1}[\Omega]$	$\geq 10^{20}$	10^{20}
(2) $r_{in2}[\Omega]$	$\geq 10^{20}$	10^{20}
(3) $r_{ctrl}[\Omega]$	$\geq 10^{20}$	10^{20}
(4) $(\phi_1 - \phi_2)_{dif}$ [deg.]	within 180 ± 0.5	180
(5) $(\phi_1 - \phi_2)_{com}$ [deg.]	within 0 ± 0.5	0
(6) A_{dif1} [dB]	within Eq. (4.15) ± 1	within Eq. (4.15) ± 0.8
(7) A_{dif2} [dB]	within $A_{dif1} \pm 0.5$	$A_{dif1} \pm 0$
(8) A_{com1} [dB]	$\leq A_{dif1} - 10$	$A_{dif1} - 10.2$
(9) A_{com2} [dB]	within $A_{com1} \pm 0.5$	$A_{com1} \pm 0$
(10) Bias deviation ΔV[V]	≤ 0.05	0.02
Supply voltage[V]	3.0	3.0

The relationship between the differential gain given in dB A_{dif} and controlling DC voltage V_C is expressed as the following equation,

$$A_{dif} = 80(V_C - V_{CQ}), \tag{4.15}$$

where V_{CQ} is a kind of the operating point. Since the gains given by Eq. (4.15) must be within a given margin of error, we utilize the convex function of error as a fitness function given by

$$f_i = \frac{1}{m}\sum_{n=1}^{m}\left(\min\left(1, \frac{2}{\left(\frac{error(n)}{margin}\right)^2 + 1}\right)\right), \tag{4.16}$$

$$error(n) = A_{dif}(n) - A_{ideal}(n), \tag{4.17}$$

where m is the number of sampling points, $A_{dif}(n)$ and $A_{ideal}(n)$ are the simulated and theoretical differential gain at the sampling point n, and $margin$ is an acceptable error from A_{ideal}, respectively. $margin$ is 1dB as described above.

Table 4.2 lists all the specifications. the fitness function f_i given by Eq. (4.8) is used for the specifications (1), (2), and (3), Eq. (4.9) for the specifications (8) and (10), and Eq. (4.10) for the specifications (4), (5), (6), (7) and (9). The total fitness F is calculated by Eq. (4.2) where $N = 10$.

Synthesis Results

Fig. 4.9 shows the dB-linear VGA designed by the proposed method. By replacing the simplified elements G_{iL} and $G_{iR}(i = 1 \sim 10)$ of Fig. 4.9(a) and optimizing the size of transistors, we obtain the MOSFET circuit given in Fig. 4.9(b) where $M_{11L}, M_{11R} \sim M_{14L}, M_{14R}$ and $M_{15} \sim M_{17}$ give the DC bias currents and $V_{1L}, V_{1R} \sim V_{7L}$, and V_{7R} are DC voltage sources to adjust the DC bias voltage. Fig. 4.10 shows the gain characteristics of the amplifier. From the figure and Table 4.2, we find that we could obtain satisfactory characteristics.

4.4.4 Comparison of Syntheses with and without Reuse of Genetic Operations

In order to confirm the impacts of reuse of genetic operations, we compare the synthesis of each circuit under the three cases: (1) without reuse of genetic operations, (2) with the reuse starting from an empty database, and (3) with the reuse of genetic operations in the database where effective genetic operations have been accumulated during the syntheses of other similar examples. Five trials were performed under each condition with two 3.2GHz Pentium IV processors. We compared the average number of generations and average simulation time to get a successful circuit, and the success rate which is defined by the ratio of the number of successful circuits and trials. The comparison is shown in Table 4.3 and the fitness transitions of the best circuit at each generation are shown in Fig. 4.11, 4.12 and 4.13.

From these results, we can see the syntheses with the reuse of genetic operations can generate a successful circuit about two times faster at a higher success rate than those without the reuse. Using the accumulated database, the synthesis can more be accelerated. In addition to the speed up of synthesis, it must be noted that the success rate becomes 100% for all three examples. We can not necessarily guarantee 100% success rate for any circuits, however, we can expect a higher success rate using the proposed method.

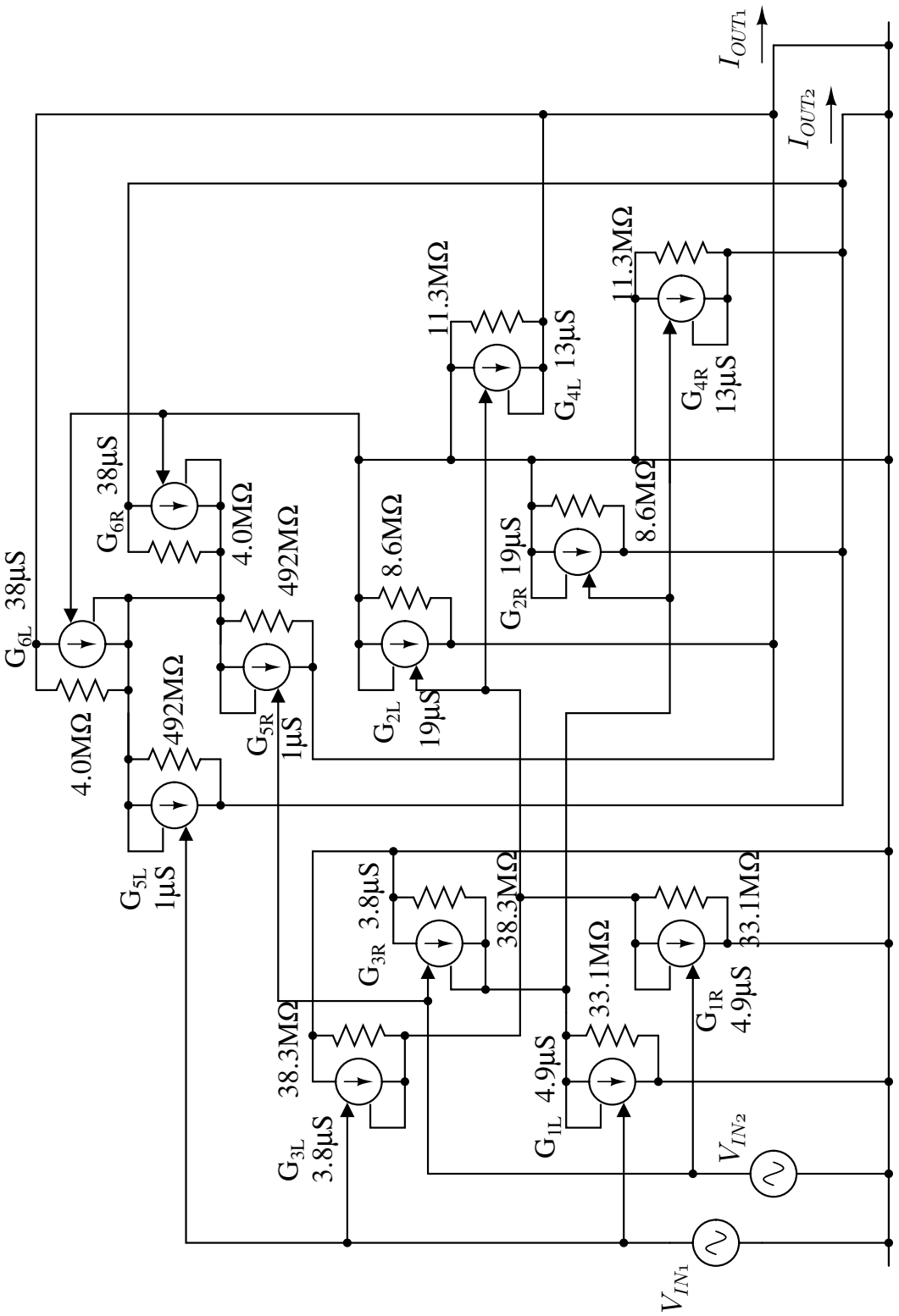
4.5 Conclusions

This chapter proposed an automated design of analog circuits accelerated by reuse of genetic operations. During the synthesis using simplified elements based on a genetic algorithm, effective genetic operations which improve circuit performances are stored in the database and reused according to circuit characteristics and topological similarities. By checking both of circuit characteristics and topological similarities,

Table 4.3: Comparison of synthesis.

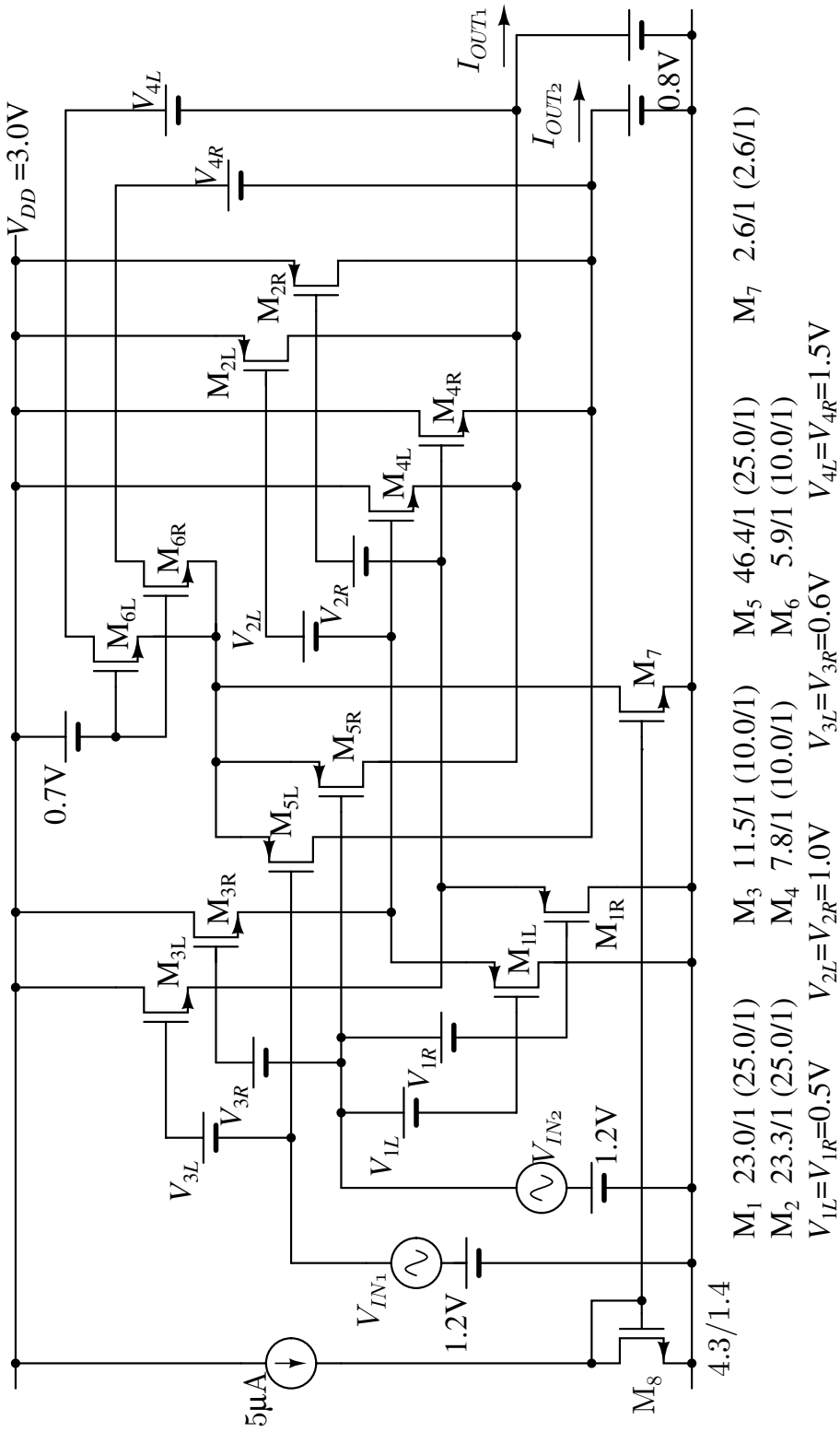
	Case	Average number of generations	Average synthesis time [sec.]	Success rate
Amplifier	(1)	37.3	4411.2	80%
	(2)	20.6	2574.9	100%
	(3)	13.8	1793.4	100%
Cubic-square	(1)	84.5	6566.4	40%
	(2)	59.2	3749.7	100%
	(3)	22.4	2121.0	100%
VGA	(1)	88.7	19443.5	60%
	(2)	66.0	12132.0	100%
	(3)	51.2	8625.4	100%

genetic operations are effectively reused and the automated design is amazingly accelerated. The design examples shows the proposed method can generate a successful circuit at 100% success rate and much faster than synthesis without the reuse.



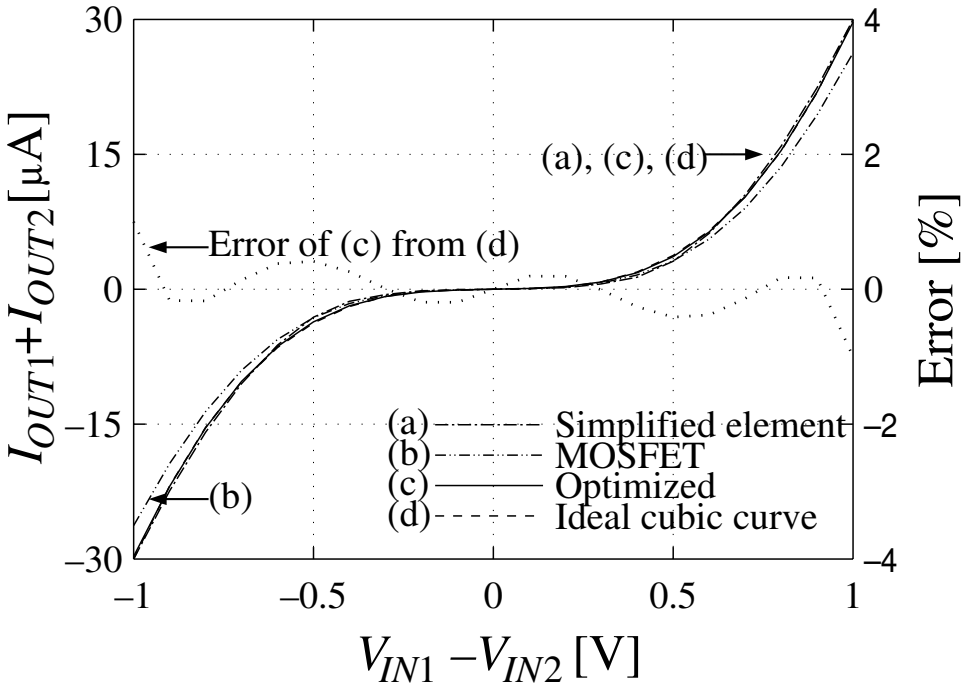
(a) Circuit with simplified elements

Figure 4.7: Synthesis result of cubing-squaring circuit

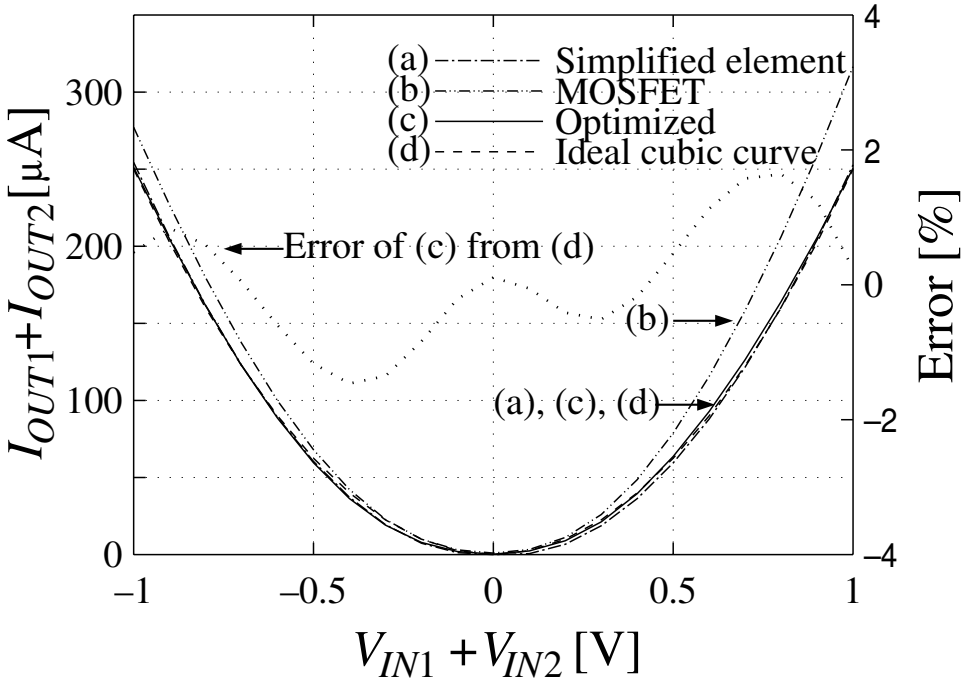


(b) Circuit replaced by MOSFETs

Figure 4.7: (Continued)

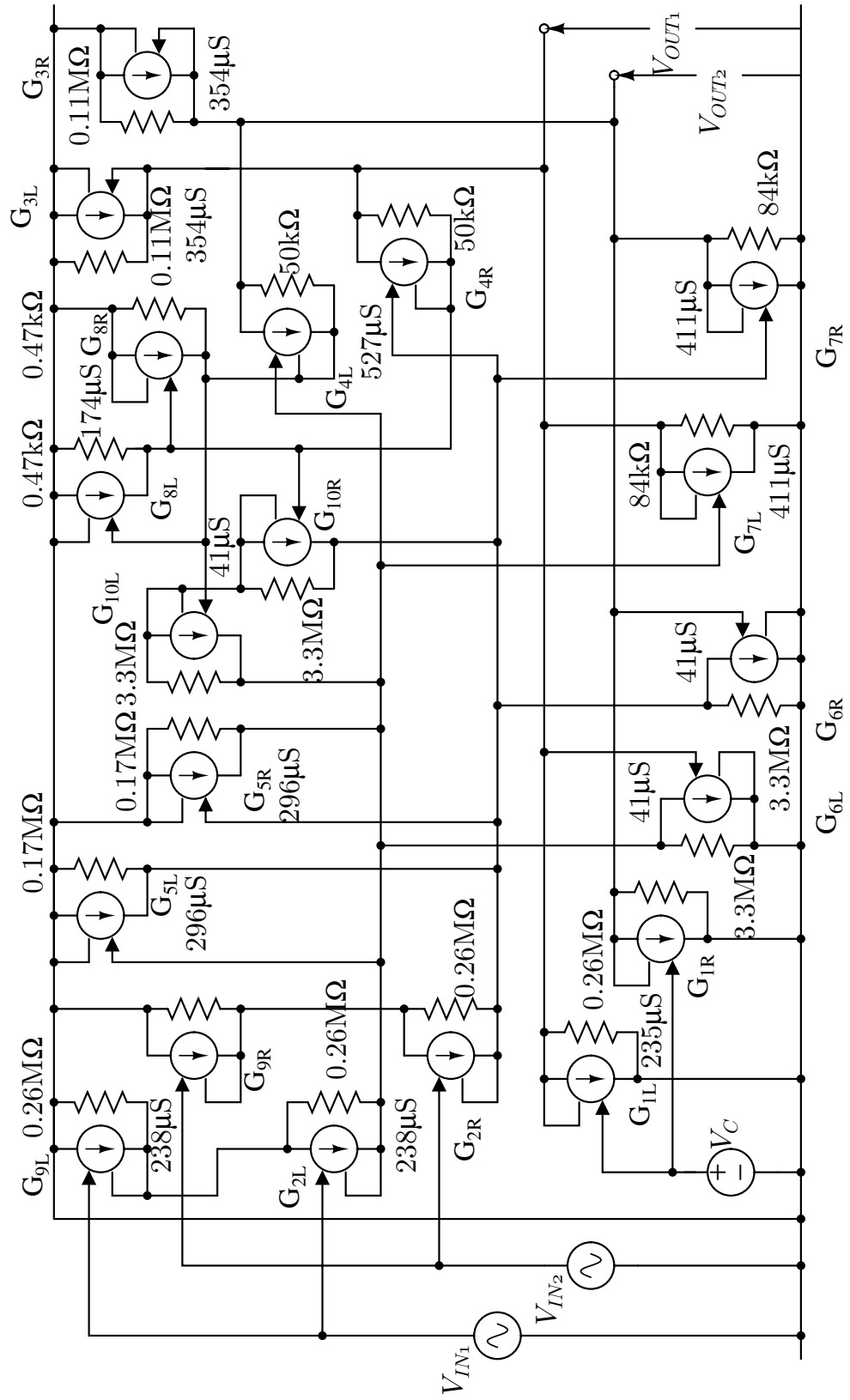


(a) Cubing in differential mode



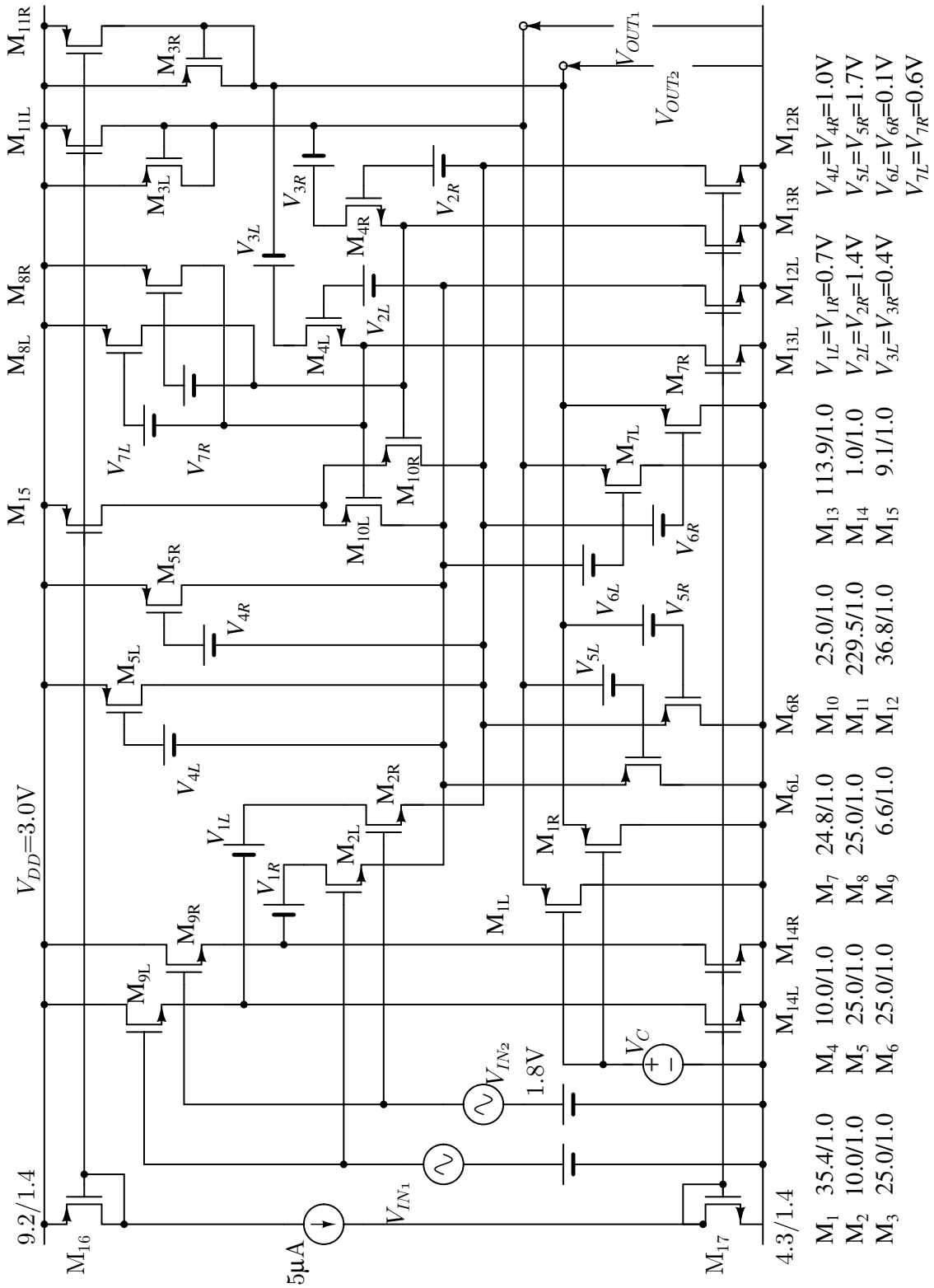
(b) Squaring in common-mode

Figure 4.8: DC characteristics of the obtained circuit



(a) Circuit with simplified elements

Figure 4.9: Synthesis result of dB-linear VGA



(b) Circuit replaced by MOSFETs

Figure 4.9: (Continued)

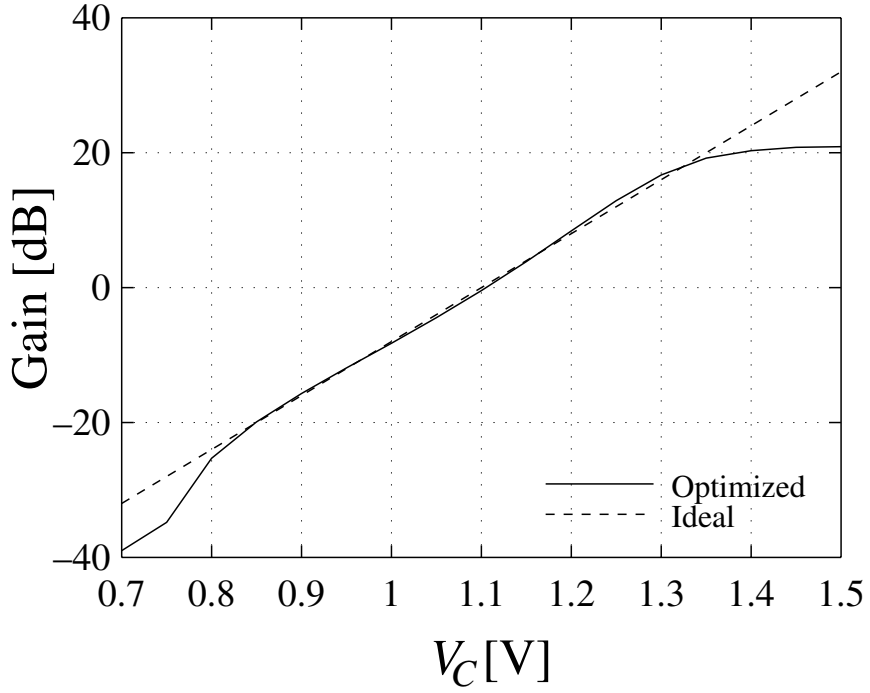
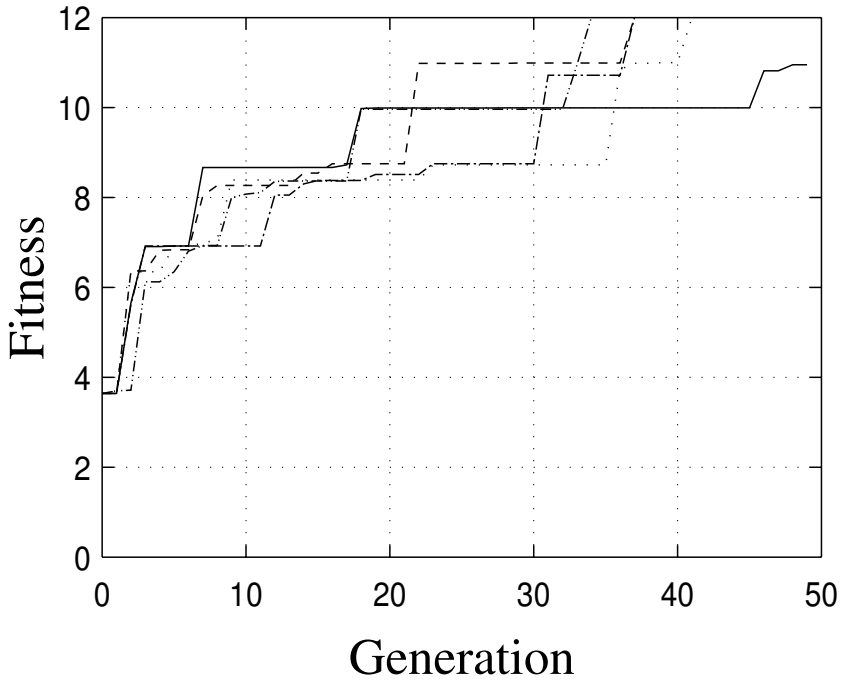
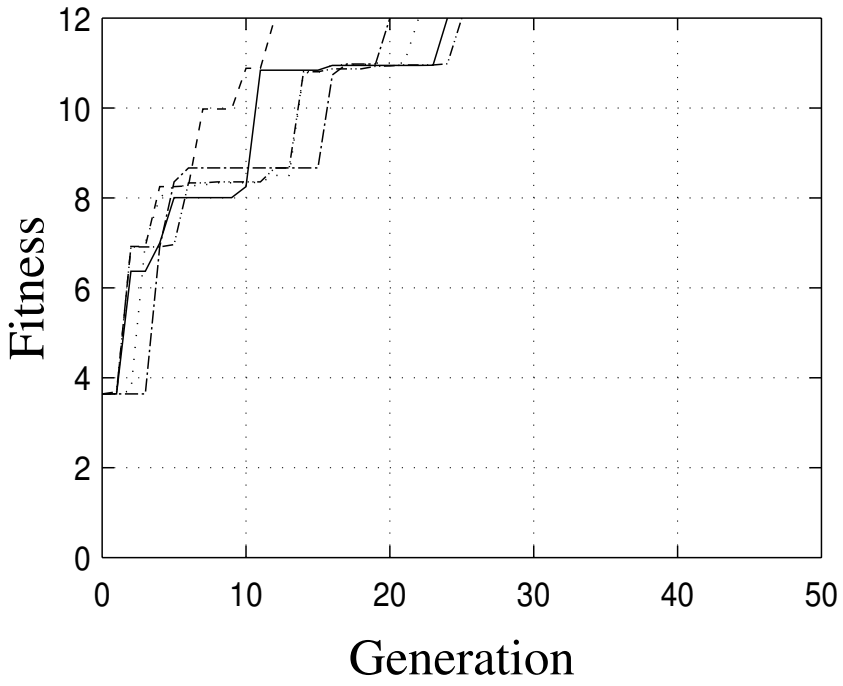


Figure 4.10: Characteristic of VGA gain versus control voltage

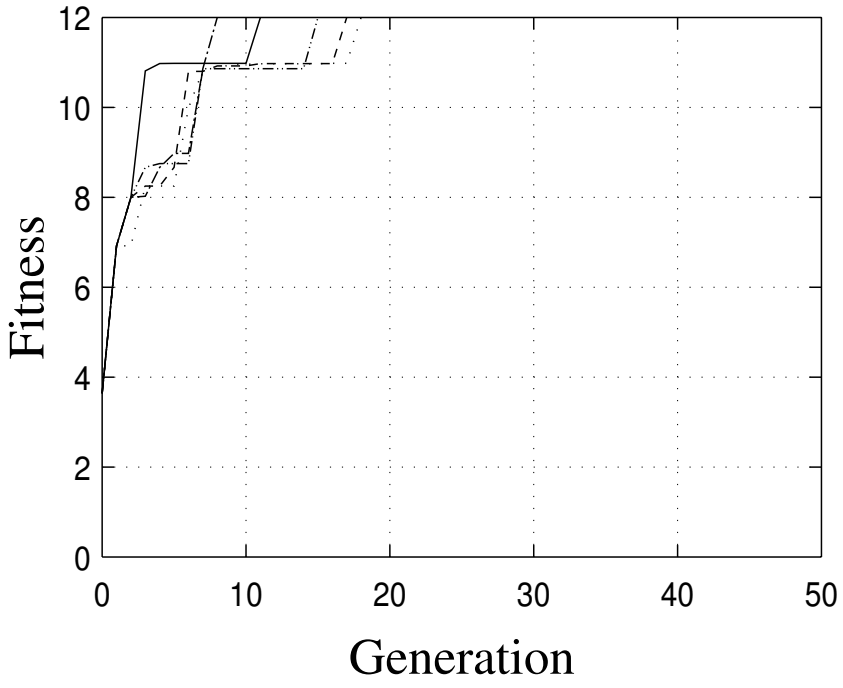


(a) Without reuse of genetic operations



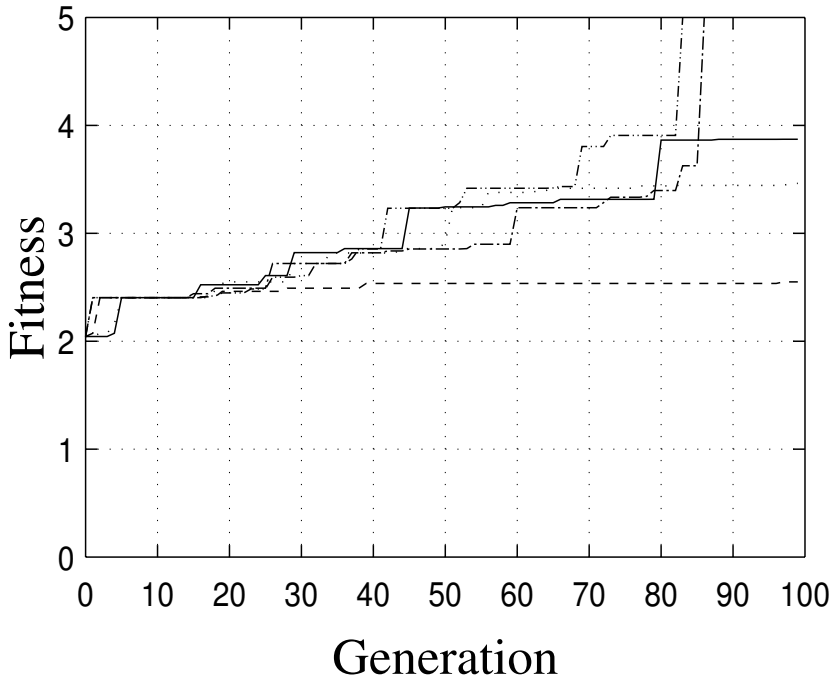
(b) With reuse of genetic operations

Figure 4.11: Fitness transition in the synthesis of voltage amplifiers



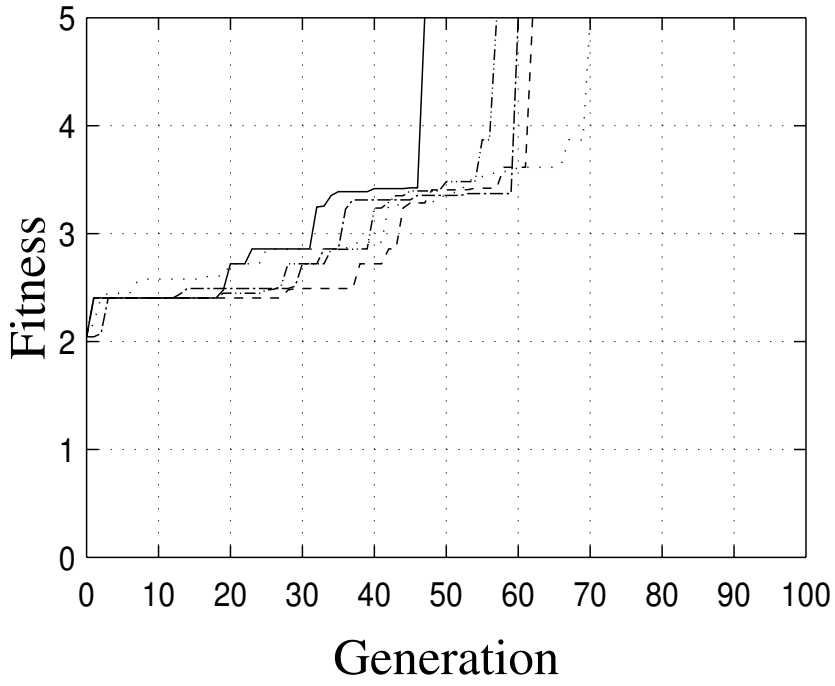
(c) With accumulated genetic operations database

Figure 4.11: (Continued)

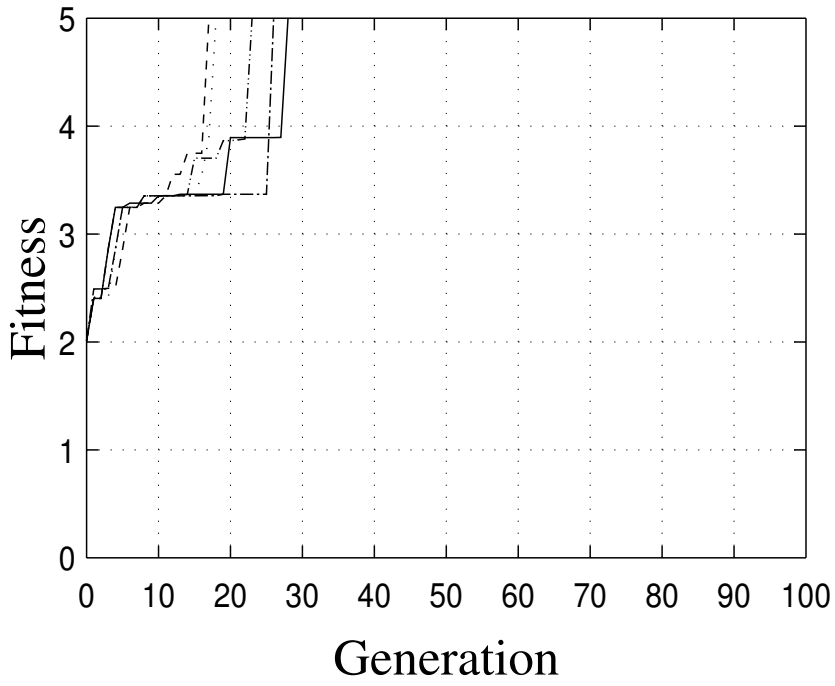


(a) Without reuse of genetic operations

Figure 4.12: Fitness transition in the synthesis of cubing-squaring circuits

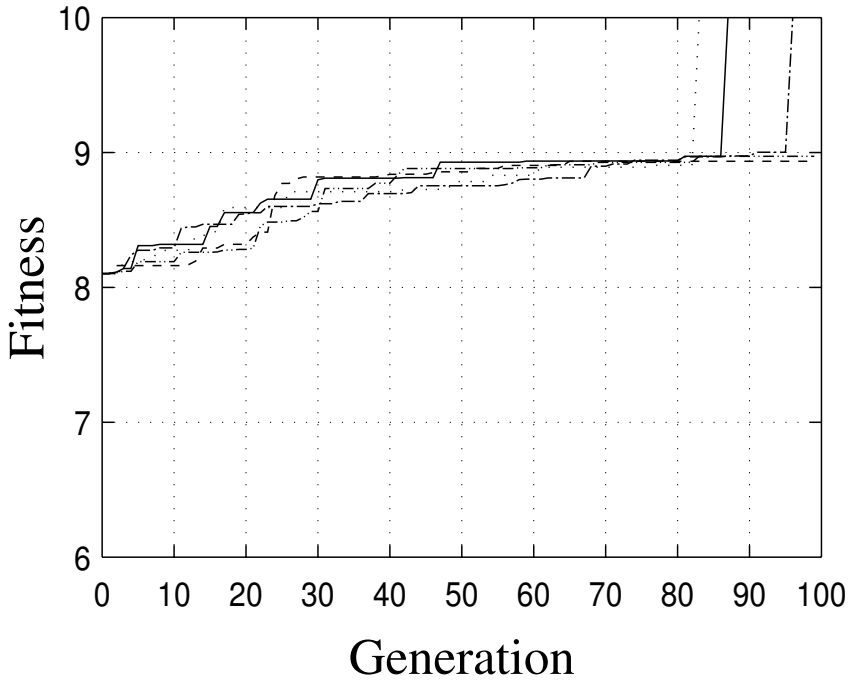


(b) With reuse genetic operations

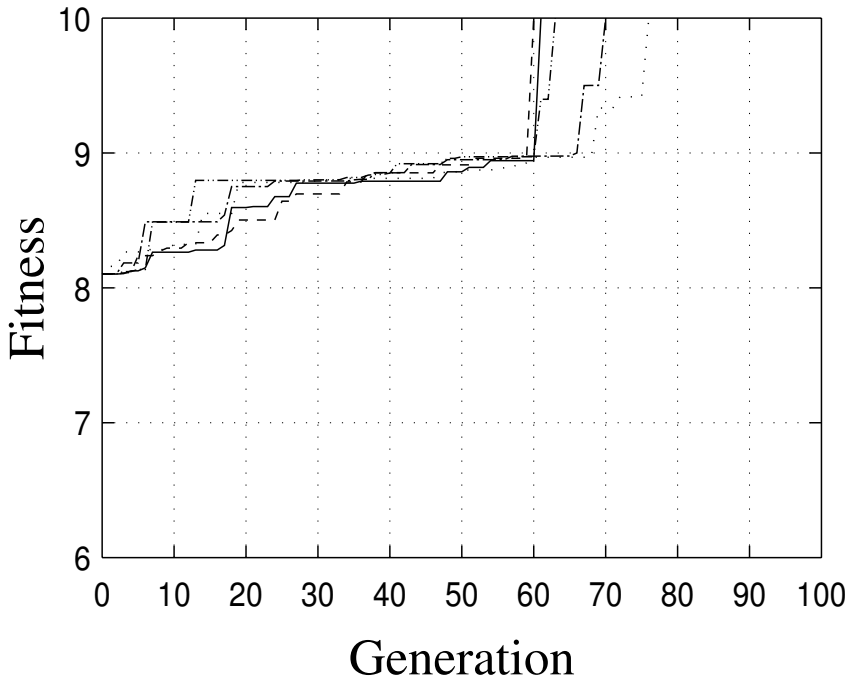


(c) With accumulated genetic operations database

Figure 4.12: (Continued)

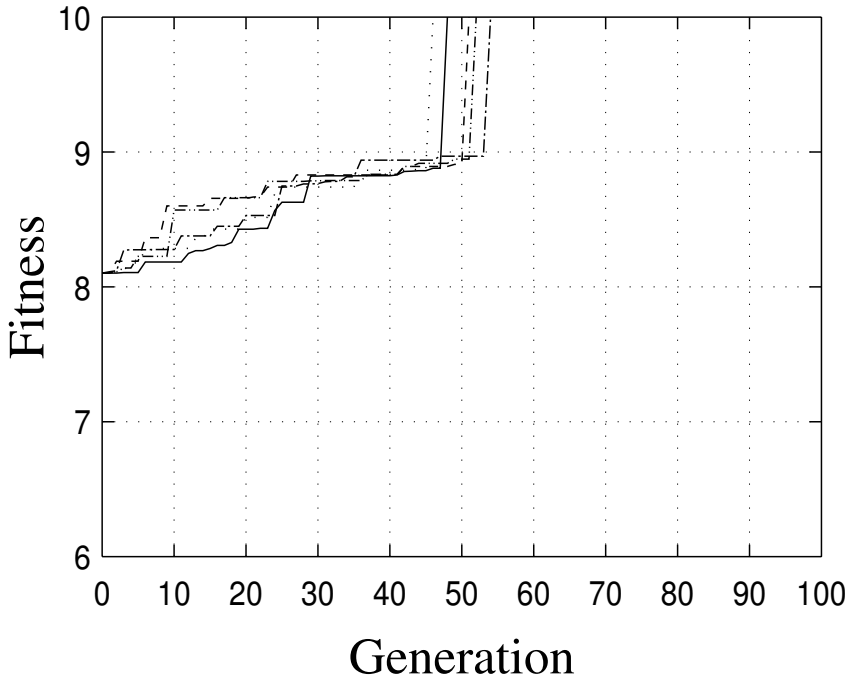


(a) Without reuse of genetic operations



(b) With reuse of genetic operations

Figure 4.13: Fitness transition in the synthesis of voltage amplifiers



(c) With accumulated genetic operations database

Figure 4.13: (Continued)

Chapter 5

Conclusions

5.1 Contributions and Conclusions of the Dissertation

In this study, the automated design of analog circuits, mainly topology synthesis, has been investigated. To generate a variety of circuits, the simulation-based optimization approach is a promising approach. However, this approach requires a huge computing time because a circuit simulator is used to evaluate the circuit performances in the synthesis process and generated circuits tend to have quite complicated structures.

In Chapter 3, an automated design of analog circuits which can extremely reduce the design time was proposed. Instead of using complicated transistor models, we introduce simplified models of transistor at the first stage of the topology synthesis. Since the simplified elements operate without bias voltage or current, this greatly shrinks the search space without limiting the circuit structure. After a circuit topology is obtained using the simplified elements, the simplified elements are replaced by transistors and the transistor parameters are optimized to meet the given specifications.

The obtained circuits only contain transistors that operate in active region and consequently the circuits become quite reasonable. Also, the proposed method is able to generate a reasonable circuit flexibly according to the given specifications. The synthesis results show the proposed method is useful for the design of both of linear and nonlinear circuits and the benchmark tests of design examples show that the proposed CAD is more than 10 times faster than the previous one. The proposed method can be universally applied to any conventional automated analog circuit design program.

Although this dissertation concentrates the discussion on CMOS analog circuits,

the proposed method can be applied to bipolar transistor circuits by using simplified elements that express the basic characteristics of bipolar transistor.

The design time is extremely reduced by introducing the simplified models of transistor, however, the synthesis still tends to be time-consuming when the number of specifications increases. Therefore, to accelerate the synthesis further, an automated design of analog circuits accelerated by reuse of genetic operations was proposed in Chapter 4. During the synthesis using simplified elements based on a genetic algorithm, effective genetic operations which improve circuit performances are stored in the database and reused according to circuit characteristics and topological similarities. This is more flexible than conventional methods. By checking both of circuit characteristics and topological similarities, genetic operations are effectively reused and the automated design is amazingly accelerated.

The effectiveness of this method was demonstrated through three design examples having complicated specifications which are quite difficult to design even by skilled analog circuit engineers. The results shows the proposed method can generate a satisfactory circuit at almost 100% success rate and much faster than synthesis without the reuse.

This study concludes that the proposed methods has pushed the automated analog circuit design much closer to commercially available level than conventional automated analog circuit design programs.

5.2 Challenges for the future

The followings are given as future tasks and directions to improve the automated analog circuit design system described in this dissertation.

- Frequency characteristics are not considered in this study. However, they are very important characteristics in the analog circuit design. One way to address this issue is to add frequency dependent elements, i.e. parasitic capacitances of the transistor to the simplified model. The size of the capacitances can be given by a similar manner to calculate the noise sources explained in Section 3.2.3. Also, the system have to deal with passive elements such as resistor and capacitor because they are frequently used in the analog circuit design for frequency compensation, amplifiers, filters, etc.

- Considering the realization of designed circuits, DFM (Design for Manufacturing) must be embedded in the system. For example, sensitivity to the process fluctuation needs to be evaluated during the synthesis.

- The proposed method in Chapter 4 stores genetic operations in a database during the synthesis. However, it must be greatly useful if we can convert the design knowledge to genetic operations by analyzing existent human-designed circuits. Therefore, an automated analysis tool to build up the genetic operation database is worth work.
- The fitness function to evaluate circuit performances needs to be improved theoretically as far as possible.

Bibliography

[1] A. A. Abidi, "RF-CMOS Comes of Age," *IEICE Trans. Electron.*, vol. E87-C, no. 6, pp. 840–853, June 2004.

[2] A. Matsuzawa, "Mixed Signal SoC Era," *IEICE Trans. Fundamentals*, vol. E87-C, no. 6, pp. 867–877, June 2004.

[3] M. G. R. Degrauwe, O. Nys, E. Dijkstra, E. A. Vittoz, S. Cserveny, C. Meixenberger, G. van del Stappen, and H. J. Oguey, "IDAC: An Interactive Design Tool for Analog CMOS Circuits," *IEEE J. Solid-State Circuits*, vol. SC-22, no. 6, pp. 1106–1115, December 1987.

[4] R. Harjani, R. A. Rutenbar, and L. R. Carley, "OASYS: A Framework for Analog Circuit Synthesis," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 8, no. 12, pp. 1247–1266, December 1989.

[5] F. El-Turky and E. E. Perry, "BLADES: An Artificial Intelligence Approach to Analog Circuit Design," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 8, no. 6, pp. 680–692, June 1989.

[6] H. Y. Koh, C. H. Séquin, and P. R. Gray, "OPASYN: A Compiler for CMOS Operational Amplifiers," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 9, no. 2, pp. 113–125, February 1990.

[7] G. G. E. Gielen, H. C. C. Walscharts, and W. M. C. Sansen, "Analog Circuit Design Optimization Based on Symbolic Simulation and Simulated Annealing," *IEEE J. Solid-State Circuits*, vol. 25, no. 3, pp. 707–713, June 1990.

[8] ——, "ISAAC: A Symbolic Simulator for Analog Integrated Circuits," *IEEE J. Solid-State Circuits*, vol. 24, no. 7, pp. 1587–1597, December 1989.

[9] M. M. Hershenson, S. P. Boyd, and T. H. Lee, "Optimal Design of a CMOS Op-Amp via Geometric Programming," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 22, no. 1, pp. 1–21, January 2001.

[10] P. Mandal and V. Visvanathan, "CMOS Op-Amp Sizing Using a Geometic Programming Formulation," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 20, no. 1, pp. 22–38, January 2001.

[11] G. Wolfe and R. Vemuri, "Extraction and Use of Neural Network Models in Automated Synthesis of Operational Amplifiers," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 22, no. 2, pp. 198–212, February 2003.

[12] 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 Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 7, no. 4, pp. 501–519, April 1988.

[13] H. Onodera, H. Kanbara, and K. Tamaru, "Operational-Amplifier Compilation with Performance Optimization," *IEEE J. Solid-State Circuits*, vol. 25, no. 2, pp. 466–473, April 1990.

[14] E. S. Ochotta, R. A. Rutenbar, and L. R. Carley, "Synthesis of High-Performance Analog Circuits in ASTRX/OBLX," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 15, no. 3, pp. 273–293, March 1996.

[15] C. R. C. D. Ranter, G. V. del Plas, M. S. J. Steyaert, G. G. E. Gielen, and W. M. C. Sansen, "CYCLONE: Automated Design and Layout of RF LC-Oscillators," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 21, no. 10, pp. 1161–1170, Octorber 2002.

[16] D. H. Horrocks and Y. M. A. Khalifa, "Genetically Evolved FDNR and Leap-Frog Filters Using Preferred Component Values," in *Proc. European Conference on Circuit Theory and Design*, 1995, pp. 359–362.

[17] R. S. Zebulum, M. A. Pacheco, and M. Vellasco, "Synthesis of CMOS Operational Amplifiers Through Genetic Algorithms," in *Proc. XI Brazilian Symposium on Integrated Circuit Design*, 1998, pp. 125–128.

[18] M. Chu and D. J. Allstot, "Elitist Nondominated Sorting Genetic Algorithm Based RF IC Optimizer," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 3, pp. 535–545, March 2005.

[19] N. Kitamura and N. Takai, "Optimizing Method for Analog Circuit Design Using Immune Algorithm," in *Papers of Technical Meeting on Electronic circuits.* IEEJ, 2005, pp. 37–42.

[20] M. Chu and D. J. Allstot, "An Elitist Distributed Particle Swarm Algorithm for RF IC Optimization," in *Proc. of the Asia and South Pacific Design Automation Conference*, 2005, pp. 671–674.

[21] M. Krasnicki, R. Phelps, R. A. Rutenbar, and L. R. Carley, "MAELSTROM: Efficient Simulation-Based Synthesis for Custom Analog Cells," in *Proc. 36th Annual Design Automation Conf.*, 1999, pp. 945–950.

[22] J. Park, K. Choi, and D. J. Allstot, "Parasitic-Aware RF Circuit Design and Optimization," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 10, pp. 1953–1966, October 2004.

[23] N. Damavandi and S. Safavi-Naeini, "A Hybrid Evolutionary Programming Method for Circuit Optimization," *IEEE Trans. Circuits and Systems-I*, vol. 52, no. 5, pp. 902–910, May 2005.

[24] 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 Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 19, no. 6, pp. 703–717, June 2000.

[25] *Circuit Explorer*, Synopsys, Inc., http://www.synopsys.com/.

[26] G. G. E. Gielen and R. A. Rutenbar, "Computer-Aided Design of Analog and Mixed-Signal Integrated Circuits," *Proc. IEEE*, vol. 88, no. 12, pp. 1825–1852, December 2000.

[27] W. Kruiskamp and D. Leenaerts, "DARWIN: Analog Circuit Synthesis Based on Genetic Algorithms," *International Journal of Circuit Theory and Applications*, vol. 23, pp. 285–296, 1995.

[28] P. C. Maulik, L. R. Carley, and R. A. Rutenbar, "Integer Programming Based Topology Selection of Cell-Level Analog Circuits," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 14, no. 4, pp. 401–412, April 1995.

[29] 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 Trans. Evol. Comput.*, vol. 1, no. 2, pp. 109–128, July 1997.

[30] J. B. Grimbleby, "Automatic analogue circuits synthesis using genetic algorithms," *IEE Proceedings of Circuits, Devices and Systems*, vol. 147, no. 6, pp. 319–323, 2000 2000.

[31] J. D. Lohn and S. P. Colombano, "A Circuit Representation Technique for Automated Circuit Design," *IEEE Trans. Evol. Comput.*, vol. 3, no. 3, pp. 205–219, September 1999.

[32] H.-S. Hou, S.-J. Chang, and Y.-K. Su, "Practical Passive Filter Synthesis Using Genetic Programming," *IEICE Trans. Electron.*, vol. E88-C, no. 6, pp. 1180–1185, June 2005.

[33] H. Shibata and N. Fujii, "An Evolutionary Synthesis of Analog Active Circuits Using Current Path Based Coding," *IEICE Trans. Fundamentals*, vol. E84-A, no. 10, pp. 2561–2568, October 2001.

[34] H. Shibata, S. Mori, and N. Fujii, "Automated Design of Analog Circuits Using a Cell-Based Structure," *IEICE Trans. Fundamentals*, vol. E86-A, no. 2, pp. 364–370, February 2003.

[35] T. Sripramong and C. Toumazou, "The Invention of CMOS Amplifiers Using Genetic Programming and Current-Flow Analysis," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 21, no. 11, pp. 1237–1252, November 2002.

[36] M. Natsui, N. Homma, T. Aoki, and T. Higuchi, "Synthesis of Current Mirror Circuits Based on Evolutionary Graph Generation," in *Proc. the 17th Workshop on Circuits and Systems in Karuizawa*, 2004, pp. 415–420.

[37] H. Shibata and N. Fujii, "Analog Circuit Synthesis Based on Reuse of Topological Features of Prototype Circuits," *IEICE Trans. Fundamentals*, vol. E84-A, no. 11, pp. 2778–2784, November 2001.

[38] J. R. Koza, M. A. Keane, and M. J. Streeter, "The Importance of Reuse and Development in Evolvable Hardware," in *Proceedings of the 2003 NASA/DoD Conference on Evolvable Hardware*, 2003, pp. 33–42.

[39] T. R. Dastidar, P. P. Chakrabarti, and P. Ray, "A Synthesis System for Analog Circuits Based on Evolutionary Search and Topological Reuse," *IEEE Trans. Evol. Comput.*, vol. 9, no. 2, pp. 211–224, April 2005.

[40] M. Murakawa, S. Yoshizawa, T. Adachi, S. Suzuki, K. Takasuka, M. Iwata, and T. Higuchi, "Analogue EHW Chip for Intermediate Frequency Filters," in *Proceedings of the Second International Conference on Evolvable Systems: From Biology to Hardware*. Springer-Verlag, 1998, pp. 134–143.

[41] T. Higuchi, M. Iwata, D. Keymeulen, H. Sakanashi, M. Murakawa, I. Kajitani, E. Takahashi, K. Toda, M. Salami, N. Kajihara, and N. Otsu, "Real-World Applications of Analog and Digital Evolvable Hardware," *IEEE Trans. Evolutionary Computation*, vol. 3, no. 3, pp. 220–235, September 1999.

[42] A. Stoica, D. Keymeulen, R. S. Zebulum, A. Thakoor, T. Daud, G. Klimeck, Y. Jin, R. Tawel, and V. Duong, "Evolution of analog circuits on Field Programmable Transistor Arrays," in *Proceedings of the Second NASA/DoD Conference on Evolvable Hardware*, 2000, pp. 99–108.

[43] A. Stoica, T. Arslan, D. Keymeulen, V. Duong, X. Guo, R. Zebulum, I. Ferguson, and T. Daud, "Evolutionary Recovery of Electronic Circuits from Radiation Induced Faults," in *Proceedings of 2004 Congress on Evolutionary Computation*, 2004, pp. 1786–1793.

[44] N. L. J. Ulder, E. H. L. Aarts, H.-J. Bandelt, P. J. M. van Laarhoven, and E. Pesch, "Genetic Local Search Algorithms for the Traveling Salesman Problem," in *Proceedings of 1st Workshop on Parallel Problem Solving from Nature*. Springer, 1991, pp. 109–116.

[45] B. Razavi, *Design of Analog CMOS Integrated Circuits*. McGraw-Hill, 2001.

[46] H. Mühlenbein, "Evolution in Time and Space - The Parallel Genetic Algorithm," in *Foundations of Genetic Algorithms*, G. J. E. Rawlins, Ed. Morgan Kaufmann Publishers, 1991, pp. 316–337.

[47] L. J. Eshelman, "The CHC Adaptive Search Algorithm: How to Have Safe Search When Engaging in Nontraditional Genetic Recombination," in *Foundations of Genetic Algorithms*, G. J. E. Rawlins, Ed. Morgan Kaufmann Publishers, 1991, pp. 265–283.

Publications Related to This Dissertation

Journal Papers

1. N. Unno, and N. Fujii, "Synthesis of Analog Circuits Starting with Idealized Elements," *IEICE Trans. Fundamentals*, vol. E89-A, no. 11, pp. 3313–3319, November 2006.

2. N. Unno, and N. Fujii, "Automated Design of Analog Circuits accelerated by Use of Simplified MOS Model and Reuse of Genetic Operations," *IEICE Trans. Electron.* (in printing).

Conference Proceedings (Referred)

1. N. Unno, and N. Fujii, "Synthesis of Analog Circuits Starting with Ideal Elements," in *Papers of Technical Meeting on Electronic circuits*, ECT-05-92, December 2005.

Conference Proceedings (Not Referred)

1. N. Unno, S. Takagi, and N. Fujii, "Design Automation of Analog Circuits by Combination of Circuit Blocks — Synthesis of OpAmp —," in *Papers of Technical Meeting on Electronic circuits*, ECT-04-18, January 2004.

2. N. Unno, N Kusakawa, and N. Fujii, "Automated Phase Compensation for Multi-stage Amplifiers," in *Papers of Technical Meeting on Electronic circuits*, ECT-04-52, June 2004.

3. N. Unno, N. Kusakawa, S. Takagi, and N. Fujii, "Automated Phase Compensation for Multi-stage Amplifiers," in *Proc. IEEE Asia-Pacific Conference on Circuits and Systems*, pp.637–640, December 2004.