2004

Multipath feedforward compensated amplifier, related dipole (doublet) compression technique, and other topics

Mark Edward Schlarmann
Iowa State University

Recommended Citation

Schlarmann, Mark Edward, "Multipath feedforward compensated amplifier, related dipole (doublet) compression technique, and other topics " (2004). *Retrospective Theses and Dissertations*. 834.
https://lib.dr.iastate.edu/rtd/834

Multipath feedforward compensated amplifier, related dipole (doublet) compression technique, and other topics

by

Mark Edward Schlarmann

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Electrical Engineering (Microelectronics)

Program of Study Committee:
Randall Geiger, Major Professor
Degang Chen
Robert Weber
Chris Chu
Dermot Hayes

Iowa State University

Ames, Iowa

2004

Graduate College
Iowa State University

This is to certify that the doctoral dissertation of

Mark Edward Schlarmann

has met the dissertation requirements of Iowa State University

Signature was redacted for privacy.

Major Professor

Signature was redacted for privacy.

For the Major Program

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

Dissertation Organization

This dissertation is organized as a collection of papers addressing three separate topics. Except for the fact that the topics are all associated with mixed-signal integrated circuit design, they are mostly unrelated. However, the contributions associated with each were significant enough to warrant their inclusion here.

The first topic addressed is a multistage multipath-compensated amplifier and related dipole compression technique. Two papers on the topic are included as Chapters 2 and 3. The first paper summarizes the progress that has been made up to this point while the second suggests a strategy for extending the existing work toward the ultimate goal of making structures composed of three or more gain stages suitable for applications that require fast settling.

The second topic presented is an interactive computer aided design (CAD) tool called *Design Space Explorer*. It was developed to assist users in manually exploring a design space. Three related documents are included as Chapters 4 thru 6. The first document is a journal paper that describes the tool. The second document is the tool's *Users Manual*. The final document describes the process of developing some specific models for use with the tool. Source code and compilation tools are also included on a supplemental CD-ROM.

The third, and final, topic describes a simple, compact CMOS transresistor. An *IEE Electronics Letter* that describes the circuit is included as Chapter 7.

The remainder of this chapter is broken into three sections each of which corresponds to one of the three topics addressed in this thesis. These sections provide introductory information and other comments in an attempt to establish the context necessary to fully appreciate the included publications.

The dissertation concludes with a chapter that briefly summarizes each of the three topics and identifies the author's contributions in each area.

Multipath Feedforward Compensation & Dipole (Doublet) Compression Technique

Background

There are many analog integrated circuits and applications that rely upon the accurate transfer of charge. Examples include A/D and D/A converters, switched-capacitor filters, and many sensor interface circuits. As an example, consider the switched-capacitor integrator shown in Figure 1. The circuit samples the input voltage onto C1 during phase 1. During phase 2, the stored charge is transferred from C1 to C2. The performance of the amplifier is critical in determining the accuracy and speed of the charge transfer. A high DC-gain is required to ensure an accurate transfer while fast-settling ensures high system throughput.

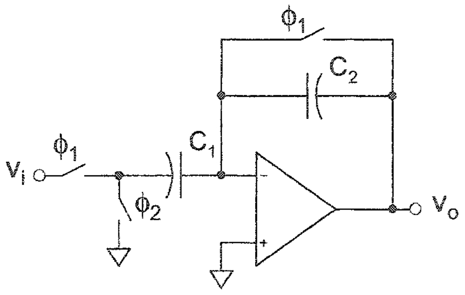


Fig. 1 Switched capacitor integrator

Because of their ability to simultaneously achieve adequate DC-gains and large gain-bandwidth (GBW) products, single stage amplifiers have traditionally been preferred for charge-transfer applications. Figure 2 shows a few of the architectures that have found widespread use. Figure 2(a) is the telescopic-cascode, 2(b) is the folded-cascode, and 2(c) is the regulated-cascode. Other variants of these structures such as the folded-regulated-cascode have found use as well.

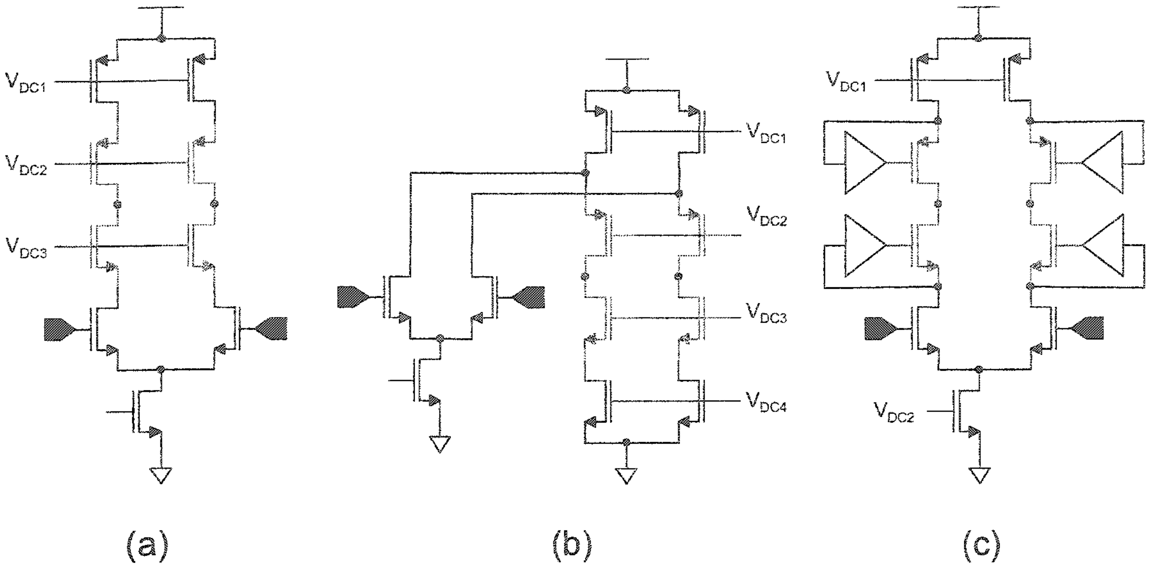


Fig. 2 Single stage amplifiers (a) telescopic-cascode, (b) folded-cascode, (c) regulated cascode

To obtain an adequate DC-gain using only a single-stage, the single-stage architectures rely on a technique termed *cascoding*. Cascoding is an output impedance enhancement technique involves vertically stacking devices between the supply rails. In Fig. 2, the cascode transistors are rendered in a lighter color than the other transistors. The problem with cascoding is that each of the devices stacked between the supply rails consumes some of the available supply range, thus the available room for output signal swing is diminished. As a result, for a fixed supply voltage, cascoded architectures have reduced output signal swing capability when compared to non-cascoded structures. Equivalently, for a fixed signal swing, cascoded output stages exhibit higher minimum supply voltages than their non-cascoded counterparts.

As supply voltages have declined, the single-stage cascoded architectures have become unsuitable for some applications because the signal swing capability is unacceptably small. As a result, designers have begun to consider alternative architectures. One example that has found use is the two-stage structure shown in Fig. 3.

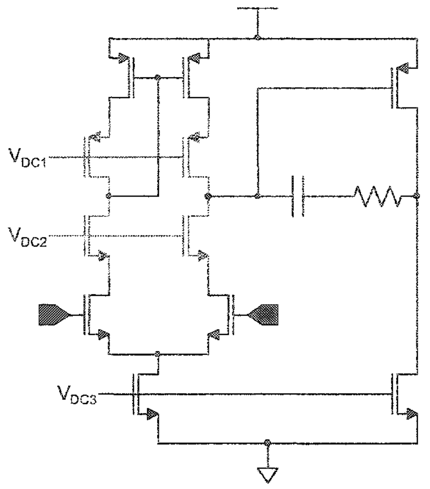


Fig. 3 Two stage amplifier

Note, in Fig. 3 that the first stage is cascoded while the second stage is not. The absence of cascode transistors in the output stage results in good output swing capability. A high DC-gain is achieved by cascoding the first stage. Signal swing is not a problem at the output of the first stage because the second stage has a gain which is typically greater than 5. As a result, the signal swing at the output of the first stage is the magnitude of the output swing scaled by the reciprocal of the gain of the second stage. e.g., if the full-scale output swing is 1V. and the gain of the second stage is 10, the swing at the output of the first stage is only 0.1V.

It needs to be mentioned that, two-stage structures, such as the one shown in Fig. 3, need to be compensated to ensure stability with negative feedback. Pole-splitting Miller compensation with a nulling resistor to flip the right half-plane zero into the left half-plane is the most commonly used technique to compensate two-stage amplifiers.

In the progression to finer process feature sizes, it is projected that supply voltages will decline even further. Furthermore, the characteristics of the active devices may degrade making the design of high-gain amplifiers even more difficult. At some point, cascoding becomes infeasible altogether. At that time, designers will be forced to switch to alternative amplifier topologies that do not rely on cascoding.

In anticipation of that moment, research into the alternative low-voltage compatible gain-enhancement techniques should be performed today. The field of potential solutions should be studied and the solutions ranked according to their performance. The goal is to develop an amplification technique that is compatible with very low supply voltages, yet it should exhibit a DC gain and settling performance that rivals the structures in use today.

Although there are many candidate techniques that need investigation, the work performed for this dissertation is an investigation of one technique that offers potential to solve the problem. The architecture under consideration is a *multistage multipath-compensated amplifier*.

Multistage Multipath-Compensated Amplifier

The multistage multipath-compensated amplifier achieves the required DC gain by cascading simple, low-voltage compatible gain stages. The term *multistage* refers to the fact that two or more stages are used to realize the amplifier. An arbitrary number of stages can be cascaded to achieve the desired DC gain.

In contrast to single-stage amplifiers, multistage amplifiers need to be *compensated* to ensure they remain stable when used in negative feedback configurations. When possible, amplifiers composed of more than two gain stages are avoided because compensating structures with three or more stages can be a tedious process. Furthermore, the compensation procedure employed may require sacrificing the available GBW product in order to ensure stability.

The compensation technique used for this amplifier involves the use of multiple feedforward paths through the amplifier. The technique is special, when compared to other compensation techniques, in the respect that a multistage amplifier can be compensated without sacrificing the achievable GBW product. Thus, in a given process, multistage multipath-compensated amplifiers can have GBW products that are just as high as their single-stage counterparts.

Due to the fact that the technique relies upon the cancellation of low-frequency pole-zero pairs (dipoles), multipath feedforward compensation has not found widespread use. Inexact cancellations result in the appearance of slow settling components in the transient response. If the magnitudes of these components are too large, they can negatively impact the settling time. A large amplifier GBW product is pointless if the settling time is determined by the unwanted slow-settling components.

Cancellation accuracy is limited due to process and environmental variations, modeling error, and device aging. In order to make the multistage multipath-compensated amplifier viable for applications that require fast settling, an amplifier topology that inherently ensures accurate dipole cancellations needs to be developed. At the present time, realization of such an amplifier appears unlikely. Alternatively, a tunable amplifier could be designed and a tuning methodology to drive the mismatch down to an acceptable level could be devised. The latter approach was pursued in this work.

Tuning Dipole Mismatch

Tuning the dipole mismatch of an *n*-stage multipath-compensated amplifier, where *n* is three or greater, is a hard problem. The following paragraphs outline some of the difficulties.

The first major difficulty involves developing a method to accurately estimate the dipoles' mismatches under the constraint that the technique must be implemented with a reasonable amount of circuitry on the CMOS die. Special test equipment such as a spectrum analyzer, digitizing oscilloscope, or precision signal sources was not available. Additionally, the tuning circuitry should not materially impact the maximum performance of the system.

One method of estimating the dipoles' mismatches involves analyzing the system's frequency response. Unfortunately, the magnitude response is not very sensitive to dipole mismatch. A closely spaced pole-zero pair is not distinguishable in the magnitude response. The phase response, however, is more sensitive. Rapid phase deviations occur in the vicinity of a mismatched dipole. The smaller the mismatch, however, the smaller the deviation in phase. Therefore, to distinguish a closely spaced dipole, accurate phase measurements are required. Another difficulty related to this approach is generating the required sinusoid of variable frequency.

Alternatively, the dipoles' mismatches can be estimated in the time-domain. This method involves analyzing the system's time-domain response to a known stimulus. Unfortunately, the transient impulse response is the superposition of decaying exponentials whose magnitudes and time constants are not precisely known. Accordingly, decomposition of a forced transient response into its constituent components is imprecise at best. Furthermore, because each additional stage added to the amplifier results in an additional component in the transient impulse response, the difficulty of this approach compounds with the number of stages in the amplifier.

An *n*-stage multipath-compensated amplifier has *(n-1)* dipoles. To tune such a structure, an amplifier with *(n-1)* control signals is required. In the ideal case, each control signal would affect only one of the dipoles and not affect the others. If this were reality, the process of tuning the dipole mismatches would be greatly simplified. In real structures, however, although the *n*'th control signal has a larger impact on the *n*'th dipole pair, all of the system poles and zeros are affected by adjusting any one of the control signals. These interdependencies complicate the tuning process because it is necessary to consider how a change in one control signal will simultaneously affect all the dipole pairs. The ability to determine how the control signals should be adjusted to reduce all of the dipole mismatches simultaneously is an important aspect of this problem.

Approach Taken

At the outset of this project, the difficulties associated with tuning the dipoles' mismatches in an amplifier with three or more gain stages seemed insurmountable. An obvious solution to the problem was not forthcoming.

The two-stage multipath-compensated amplifier is the easiest case to tune because it has a single low-frequency dipole. The tuning difficulty rapidly compounds with the number of stages in the amplifier because each additional stage introduces another dipole in the system transfer function. Therefore, it was decided to focus on the two-stage case first. It was hoped that what was learned in solving the two-stage case would help lead to a solution of the more complicated problem of tuning an amplifier with three or more stages.

The paper, presented as Chapter 2, describes a two-stage multipath-compensated amplifier and a foreground technique used to tune its dipole mismatch. The method involves sampling the transient step response of the device under test at different points in time to determine the polarity of the dipole mismatch. Using the mismatch information, an amplifier bias current is adjusted to reduce the mismatch. The entire process is repeated a large enough number of times to ensure convergence for the expected worst-case mismatch. To minimize the impact on performance, a time-domain averaging technique was employed. A prototype was integrated in a 0.25μ CMOS process and the concept was experimentally validated. The full details are in the paper, included as Chaper 2.

As was hoped for, the insight gained while developing a technique to tune a two-stage multistage multipath-compensated amplifier, led to the development of a strategy for tuning an amplifier with three or more stages. The technique is closely related to the one used to tune the two-stage structure. However, its implementation is more involved. For an *n*-stage amplifier, the method involves sampling the transient step response at *n* different points in time. The difficulty associated with determining how the (*n-1*) control signals should be adjusted to simultaneously reduce all the dipoles' mismatches is addressed by the use of the Newton-Raphson iteration. The derivatives required by the algorithm can be obtained via a finite difference method or by sampling the impulse response. Due to the complexity associated with the tuning algorithm, implementing this technique in hardware will most likely require a digital signal processor in the tuning loop. The approach is outlined in a paper contained as Chapter 3. Reduction of this concept to practice is left as future work.

Design Space Explorer

In the context of circuit design, design optimization usually involves writing mathematical expressions for the relevant performance parameters in terms of the design's degrees of freedom (DOF). If these expressions are simple enough, tractable analytical expressions for the design tradeoffs can be derived and optimization is straightforward.

In most cases, however, the expressions for the design tradeoffs are not tractable. The usual approach, in such a situation, is to formulate a cost function and use a mathematical optimization tool to perform the optimization.

At the present time, computers are reasonably good at repetitive tasks such as optimization, but they are not good at tasks that require creativity. For this reason, humans are still required to design new or improved circuit topologies.

Because of their fully-automated, non-interactive nature, the use mathematical optimization tools often have the undesired side-effect of impeding the assimilation of designer knowledge. The black-box type of interface to the optimization tool isolates the designer from the optimization process. Except for the final result, very little information is fed back to the user. The net effect is a slower rate of design knowledge assimilation which may lengthen the time required to develop new and improved circuit topologies.

Rather than being completely isolated from the optimization process, designers can more rapidly deepen their understanding of the available design tradeoffs by interactively exploring a design space. The additional insight into design performance gained by manual exploration of the design space will aid the designer in creating improved design topologies in the future.

Design Space Explorer (DSE) is a CAD tool designed to make it easy to interactively explore a circuit design space of pre-characterized circuit topologies. The interactive nature of the tool helps designers more quickly gain a qualitative understanding of how a design's degrees of freedom relate to its performance parameters and the design tradeoffs that are possible for a given design topology. Although the focus here is on circuit optimization, care was taken in the design of the application to ensure the software is suitable for design exploration in other disciplines as well. A paper that more fully describes the tool is included as Chapter 4.

DSE is fully programmable. Users can extend the application to include new or custom circuit topologies without recompiling the application. They do so by writing their own design specification files. Programming the application is documented in the DSE Users Manual which is included in this document as Chapter 5.

The application is coded in *Java*[1] and is compatible with the emerging network-centric computing paradigm. Of the benefits that accrue due to the use of the network-centric computing paradigm, the most significant is the enhanced communication of design knowledge and prevention of reinvention enabled by the use of a centralized *Design Knowledge Repository*.

For those interested in learning how to develop their own DSE models, Chapter 6 contains a document that provides a comprehensive description of how detailed models of a standard differential amplifier, a telescopic cascode, and a two-stage amplifier were developed.

Since the printed version of the source code is too long to include in this document (53677 lines of code), it has been included in electronic form on a supplemental CD-ROM.

MOS Transresistor

Many applications require converting signal currents into signal voltages. In most situations, resistors are used to perform the conversions.

Due to the low sheet resistance of polysilicon, large integrated polysilicon resistors consume a lot of die area. For this reason, polysilicon resistors are avoided for applications that are extremely cost-sensitive. High volume parts are examples of such applications. Saving just a few cents may make the difference between a product being successful in the marketplace or not. Additionally, polysilicon resistors are not well suited for applications that require massive arrays of repeated blocks that contain resistors. Examples of such circuits are neural network and sensor arrays. The area required by the resistors restricts the maximum number of elements in the array. Thus, there are many applications would benefit from a more area-efficient methods of converting the signal currents into voltages.

In this work, we report a simple MOS transresistor that is perfect for such applications. Compared to other transresistors that have appeared in the literature, the newly proposed structure offers comparable linearity while requiring less area.

[1] Java is a Trademark of Sun Microsystems, Inc.

The paper published in *IEE Electronics Letters* describes the circuit and compares its performance against several other popular transresistors. Experimental results are presented as well. The paper is included in this document as Chapter 7.

CHAPTER 2. MULTIPATH FEEDFORWARD COMPENSATED AMPLIFIER AND DIPOLE (DOUBLET) COMPRESSION TECHNIQUE

A paper to be submitted to the Journal of Solid-Stage Circuits

Mark E. Schlarmann and Randall L. Geiger

Abstract

Multipath feedforward compensated amplifiers are suitable for low-voltage applications and capable of achieving very high gains in modern deep-submicron CMOS processes. These architectures are often plagued by slow-settling due to imperfectly cancelled low-frequency pole-zero pairs. A new pole-zero mismatch compression methodology is utilized to tune the transient performance of a two-stage amplifier. The approach was validated with a prototype fabricated in a 0.25µ CMOS process.

Introduction

Competitive pressures are compelling the integration of entire systems on a chip (SOC). For mixed-signal systems, this requires that both analog and digital functionality be integrated onto the same die. The starting point for such an endeavor is generally a CMOS digital process with optional processing steps that are added to obtain analog friendly features such as low V_T transistors, temperature stable resistors, linear capacitors, etc. Although it would make the life of an analog designer easier if each of the additional process options were automatically included, each one adds to the final cost of fabricating the chip. Therefore, only those extra processing options that are absolutely required for a given application are incorporated into the production process for that application. For high-volume parts, significant efforts are expended to develop solutions that provide the required analog functionality in purely digital processes.

To achieve the levels of integration required by complex SOCs, modern deep-submicron processes are required. The digital core of a typical 0.13µm process operates at 1.2V. New and emerging processes will operate at even lower voltages [1]. Where possible, the analog circuitry needs to operate at the same low voltage as the digital core.

In contrast to digital circuit performance, analog circuits do not benefit from supply voltage reductions. Many traditional analog circuit architectures either suffer a significant degradation in performance or become completely non-functional as the supply voltage is reduced. In this paper, attention is focused on one essential analog building-block, the amplifier. Specifically, we will concentrate on an alternative low-voltage compatible transconductance amplifier intended for use in charge-transfer applications.

Integrated applications that require accurate charge transfer are abundant. Examples include switched capacitor filters, analog to digital converters, digital to analog converters, and many sensor interface circuits. In these circuits, the DC gain of the amplifier determines the precision of the charge transfer while the amplifier settling time determines the maximum system throughput.

Of the many different amplifier topologies that are capable of producing the required DC gains, *cascoded* single-stage architectures were used almost exclusively in the past. They were preferred because, in addition to meeting the gain requirements, they achieved faster settling than other structures. This is a consequence of the fact that cascoded single-stage amplifiers, in general, do not need compensation. As such, they do not suffer a bandwidth reduction due to the process of compensation.

The viability of single-stage cascoded structures is declining as fabrication technology progresses to lower supply voltages. This problem is due to the fact that *cascoding* involves stacking several devices between the supply rails. As a result, cascoded gain stages require a larger minimum supply voltage to operate than non-cascoded stages. In emerging digital processes, the signal swing capability of the cascoded amplifiers will not be large enough to meet the performance requirements of most mixed-signal applications.

As a transitionary step in the evolution to much lower supply voltages, one might consider using an existing amplifier architecture and simply reduce the maximum signal swing to the match the capability of the amplifier. Unfortunately, this is usually not a viable option. For example, in order to maintain the same signal to thermal-noise ratio on the sampling capacitors when the maximum signal swing is reduced by a factor of 2, the size of the hold capacitors must be increased by a factor of 4. In such a scenario, to maintain the same system throughput, the transconductance gain of the amplifier has to also increase by a factor of 4, which requires a 16-fold increase in the quiescent current of the amplifier. From this example, one can see that the amplifier power consumption is very sensitive to signal swing.

Alternative amplifier architectures that are compatible with lower minimum supply voltages and that are capable of operating at speeds comparable to what is achievable with the amplifiers in use today need to be developed. In the following section, we will briefly survey a few of the low-voltage amplifier design techniques that have appeared in the literature and then focus on a new approach.

Replica Gain Enhancement

Replica-amp gain enhancement [2] is an amplifier gain-enhancement technique that offers potential for low-voltage implementation. The technique uses two transconductance amplifiers; one termed the *main* amplifier and the other the *replica* amplifier. The replica amplifier generates a signal-dependent current that is injected into the output node of the main amplifier. Since the replica amp supplies a portion of the current required to drive the main amplifier's load, less current is drawn from the main amplifier. As a result, the magnitude of the main amplifier's error voltage is reduced which is analogous to increasing the gain of the main amplifier. The matching accuracy of the replica and main amplifiers largely determines the amount of gain enhancement, which is slightly disadvantageous because in the absence of good process characterization data, it is difficult to make an a priori prediction of the gain. The constituent amplifiers can be relatively simple low-voltage compatible gain stages. Therefore, this technique offers good potential for operating at high speeds in deep submicron processes. Concerns about the possibility of slow settling components appearing in the transient response have been addressed in [3]. The ultimate performance potential of this technique has not yet been demonstrated.

Positive Feedback

Due to concerns regarding system stability, performance variations, noise magnification, and yield predictability, many in the design community have acquired negative perceptions regarding the use of positive feedback as a method for enhancing the DC gain of an amplifier. As a result, the use of positive feedback as a low-voltage compatible gain-enhancement technique has not been fully explored. In reality, embedding a positive-feedback amplifier within a larger negative-feedback system does offer significant potential for low-voltage applications and needs fuller investigation [4].

Circuit Level Solutions

Circuit techniques may circumvent the need for extremely high-gain amplifiers in charge transfer circuits altogether. Consider the class of circuits that employ correlated double sampling techniques [5]. If successive samples are correlated, these circuits exhibit an effective gain that is larger than the actual gain of the amplifier. For fully correlated samples, the gain is effectively squared facilitating the use of simple, low-gain, low-voltage compatible architectures for the amplifier. For Nyquist converters, successive samples are not necessarily correlated. By fixing the input for two cycles, a Nyquist rate converter can be constructed at the cost of halving the throughput. Other circuit-level solutions to circumvent the need for high-gain amplifiers in charge transfer applications may also emerge.

Cascaded Structures

Cascading several low-complexity gain stages to achieve a high-gain amplifier is another technique that is compatible with low-voltage supplies. Until now, however, amplifiers with more than two stages have found very limited adoption. The extra design effort and circuit complexity has not yet been justifiable. Supply voltage reductions and degradation of the quality of the active devices in deep-submicron processes may render the gain of the single and two-stage structures inadequate requiring three or more gain stages in the near future.

To ensure their stability in feedback configurations, multistage amplifiers need to be *compensated.* Popular compensation techniques for two-stage amplifiers are based upon dominant pole compensation generally in the form of pole-splitting (Miller) compensation. Feedforward compensation [6] finds occasional use. More sophisticated compensation techniques are required for amplifiers with three or more stages.

Examples of *n*-stage compensation techniques include nested Miller compensation [7,8,9], hybrid nested Miller compensation [7], multipath nested Miller compensation [7], nested Gm-C compensation [10], nested Miller compensation with feedforward g_m stage [12], damping factor control frequency compensation [11], active feedback frequency correction [13], and embedded RC frequency compensation [14]. Good comparative studies of these structures are found in [12-13].

Beyond ensuring an adequate phase margin, amplifier compensation techniques are characterized by the amount of gain they provide and the maximum bandwidth they can achieve. For high-speed applications, a fast-settling transient response is also important.

Many of the multistage amplifier compensation schemes utilize some variant of the multipath feedforward architecture. The term *multipath* signifies the presence of multiple signal paths through the amplifier while *feedforward* indicates the corresponding signals propagate in one direction only, from the input towards the output. Many of the multistage compensation techniques suffer significant reductions in the obtainable bandwidth for each stage added to the amplifier. This is somewhat ironic since multipath feedforward topologies were conceived as a way to compensate for bandwidth reduction [7].

Thomsen et al [15,16] used a multipath feedforward architecture to realize a conditionally stable 5-stage amplifier. It was not suitable for applications that required fast settling because of the presence of low frequency dipoles (closely spaced pole-zero pairs are often referred to as *doublets* but are more aptly described by the term *dipoles*). In [17] it was shown how an architecture similar to Thomsen's could be used to realize an unconditionally stable *n*-stage amplifier. The technique offers the potential to realize amplifiers that can simultaneously achieve high gain-bandwidth products and large DC gains. However, these structures still rely on pole-zero cancellations to prevent the appearance of slow-settling components in the transient response. Thandri et al. [18] observed that for a two-stage structure the cancellation accuracy required to obtain a certain level of settling performance is relaxed as the dipole is positioned closer and closer to the UGF of the amplifier (refer Fig. 11 of [19]). By positioning the dipole at as high a frequency as possible, they implemented an amplifier whose sensitivity to dipole mismatch was lower than one with its dipole positioned at a lower frequency. Although viable for two-stage structures, this concept is not extendible to amplifiers with a larger number of stages because amplifiers with more than two stages require spacing the dipoles at successively lower frequencies. Furthermore, because the cancellation is inexact, the slow-settling components are not suppressed. As a result, their method is most suitable for applications that require low to medium resolution settling accuracies. Practical implementations of [17] that maintain accurate pole-zero cancellation over process and environmental variations have not appeared in the literature.

In this work, a calibration technique that compresses the dipole mismatch in a multi-stage multipath feedforward compensated amplifier is reported. Experimental results for a two-stage structure are presented to validate the approach

Although the pole-zero calibration technique is presented here in the context of a multipath feedforward architecture, the calibration technique and the architectural issues are separable. The technique can be used to calibrate other amplifier topologies that suffer from slow-settling due to low-frequency dipoles as well.

A description of the multipath feedforward compensation architecture is presented in Section II. The new calibration technique is introduced in Section III and experimental results validating the technique are included in Section IV.

Multipath Feedforward Compensation

Using *multipath feedforward* compensation, an *n*-stage amplifier can be compensated to ensure that it is unconditionally stable when used with attenuative negative feedback. The technique involves a nested iteration of the module shown in Fig. 4. Each of the gain blocks in the figure represents a gain stage with a first-order transfer function of the form:

$$H_k(s) = \frac{A_k}{1 - s/P_k} \qquad k \in \{ a, b \} \tag{1}$$

where A_k and P_k are the DC gain and pole location respectively.

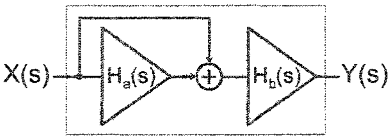


Fig. 4 Basic multipath feedforward module

The transfer function of the basic module is given by:

$$H_{ab}(s) = \frac{Y(s)}{X(s)} = H_b(s)[1 + H_a(s)] \tag{2}$$

Substituting (1) into (2) for $H_a(s)$ and $H_b(s)$ and simplifying results in:

$$H_{ab}(s) = \frac{A_b(1 + A_a)\left(1 - \frac{s}{P_a(1 + A_a)}\right)}{(1 - s/P_a)(1 - s/P_b)} \tag{3}$$

Adjusting the bandwidth of the second stage to satisfy:

$$P_b = P_a(1 + A_a) = P_a + GBW_a \tag{4}$$

results in pole-zero cancellation and (3) simplifies to:

$$H_{ab}(s) = \frac{A_b(1 + A_a)}{1 - s/P_a} \tag{5}$$

Thus, if the condition of (4) is satisfied, the basic module exhibits a first-order response with a DC gain, pole location, and gain-bandwidth product (GB) given by:

$$A_{ab} = A_b(1 + A_a) \tag{6}$$

$$P_{ab} = P_a \tag{7}$$

$$GB_{ab} = A_b P_a (1 + A_a) = A_b P_b = GB_b \tag{8}$$

Since the DC gain is proportional to the product of the gains of the individual stages, the overall gain is enhanced if each stage has a DC gain that is larger than 1. The first stage determines the pole location while the second stage determines the overall GB. Since the GB of the module is equal to the GB of its second stage, the module is capable of achieving operating speeds that are just as fast as a single stage. Unlike many other compensation techniques, multipath feedforward compensated amplifiers are not restricted to operating speeds that are lower than what a single-stage amp can achieve.

By nesting the basic module within itself, amplifiers with more than two stages can be created. The procedure used to design a voltage amplifier is illustrated in Fig. 5. Two simple first-order gain stages are used to realize the basic module as shown in Fig. 5(a). The bandwidth of the second stage is adjusted to ensure that pole-zero cancellation occurs. Since the resultant structure has a first-order transfer function, it can be thought of as equivalent to a single stage and can be nested within another module as shown in Fig. 5(b). The pole-zero cancellation process is repeated again for the module of Fig. 5(b) resulting in a three-stage amplifier with a transfer function of:

$$H_{123}(s) = \frac{V_3(s)}{V_i(s)} = \frac{A_3(1 + A_2(1 + A_1))}{1 - s / P_1} \tag{9}$$

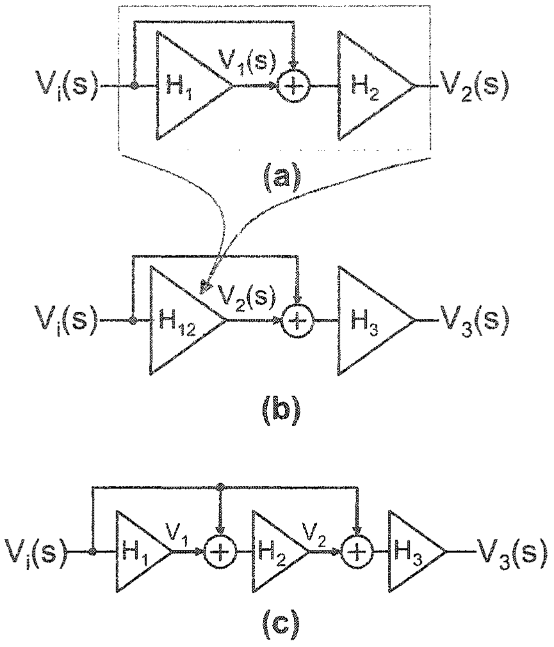


Fig. 5 Multipath feedforward compensation technique (a) basic module, (b) nesting the module within another, (c) amplifier of (b) *flattened* to yield the resultant three-stage amplifier

Thus, the overall structure has a first-order transfer function even though it consists of three stages. If each stage has a gain that is much larger than one, the DC gain can be quite large. The overall GB is identical to the GB of the final stage. The resultant amplifier is shown in *flattened* form in Fig. 5(c). Amplifiers with more than three stages can be constructed by additional nesting.

Fig. 6 is a plot of the typical closed-loop pole and zero locations for a third-order multipath feedforward compensated amplifier used in a standard feedback configuration. If perfect cancellation occurred, the system would be exactly first-order. Practically, mismatches always exist and the system only approximates a first-order response.

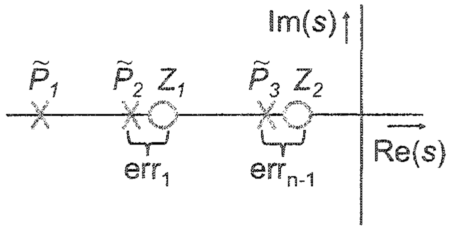


Fig. 6 Closed-loop pole and zero locations for a third-order multipath feedforward compensated amplifier

Imperfect cancellation of closely-spaced pole-zero pairs results in the presence of extra decaying exponential components in the transient response [20-22]. These additional components decay more slowly than the desired component because the poles responsible for their appearance lie at lower frequencies than the *uncovered* pole.

If the magnitudes of the mismatches are small, the magnitudes of the undesirable components are also small. However, if the mismatches are large, the sizes of the undesired components are also large. Unfortunately, accurate pole-zero cancellation is not easy to ensure due to modeling errors, process variation, and environmental factors. For this reason, for applications that require fast settling, amplifiers with low frequency dipoles are normally avoided. To extend the applicability of the proposed structures to applications that require fast settling, a calibration technique that compresses the dipole mismatch was developed. The remainder of this paper is devoted to the issue of the calibration of dipole mismatch.

Calibration Technique

The material here will be restricted to a self-calibration technique for a two-stage multipath feedforward compensated amplifier. The more difficult problem of calibrating amplifiers with more than two stages is beyond the scope of this paper but some preliminary results addressing the more general problem can be found in [23].

The switched capacitor gain stage shown in Fig. 7 was constructed with the two-stage multipath feedforward compensated amplifier shown in Fig. 8. During the charge transfer phase, ϕ_2, this configuration has a pole-zero constellation similar to that shown in Fig. 9. If the low-frequency pole-zero pair cancels exactly, the system exhibits a first-order step-response like that shown in Case B of Fig. 10. In that case, the response is a standard exponential rise from zero to the final steady-state output value.

Cases *A* and *C* of Fig. 10 illustrate the effect of pole-zero mismatch on the step-response. Cases *A* and *C* correspond to the scenarios where the low-frequency pole lies on the real axis to the left and to the right of the zero respectively. Notice that when the low-frequency pole lies to the left of the zero, overshoot occurs in the step response, whereas when the pole lies to the right of the zero, undershoot occurs. The sensitivity of the step response to the dipole mismatch can be exploited to obtain a signal that indicates the polarity of the mismatch. This error signal is used to calibrate the amplifier.

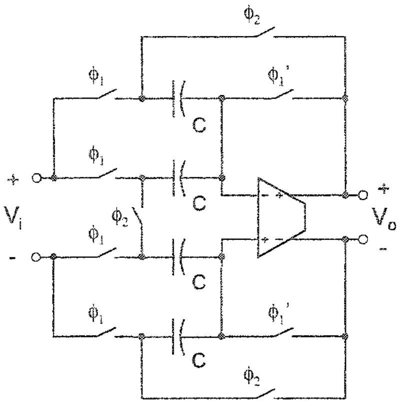


Fig. 7 Switched capacitor gain stage schematic

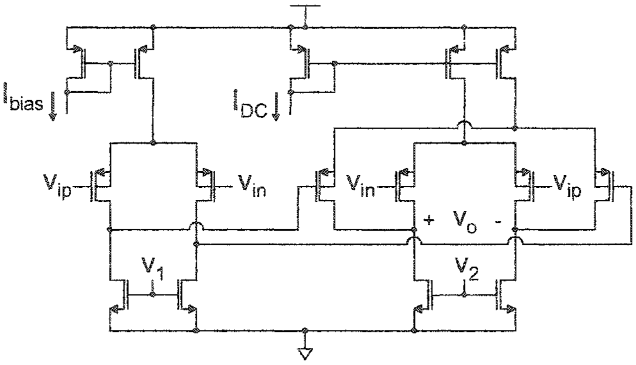


Fig. 8 Schematic diagram of the multipath feedforward compensated two-stage amplifier used in the switched capacitor gain stage of Fig. 7

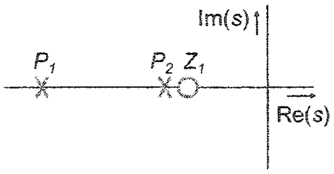


Fig. 9 Pole-zero constellation during charge transfer phase

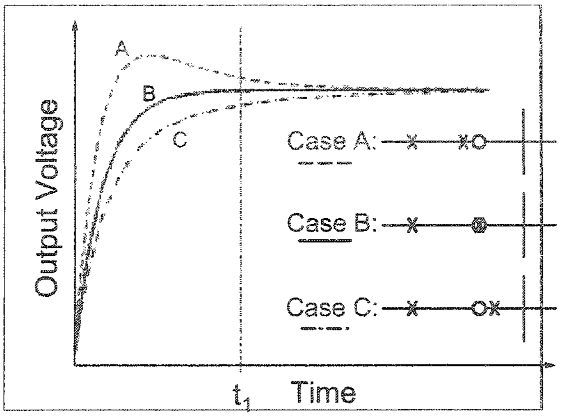


Fig. 10 Step responses for two-stage multipath amplifiers with varying degrees of pole-zero mismatch

The technique involves sensing the slope of the step response after a delay that is several times longer than the time constant of the high-frequency pole but still shorter than the time constant of the low-frequency pole. For example, consider the value of the derivatives of the responses shown in Fig. 10 at time t_1. As illustrated in the figure, a negative slope indicates that the pole lies to the left of the zero while a positive slope indicates the pole lies to the right of the zero. In the proposed calibration algorithm, the mismatch information is used to adjust the amplifier bias current, I_{bias}, which shifts the locations of the dominant pole in the direction needed to cause pole-zero cancellation. The measure/adjust process is repeated until the required cancellation accuracy is achieved.

Slope Measurement

One method for sensing the slope is to use a RC-based differentiator circuit. Although straightforward to implement, this method does not work well in practice because the signals we are trying to differentiate are too fast. The RC differentiator exhibits a sensitivity/speed tradeoff. Reducing the time constant of the differentiator increases its bandwidth and improves its ability to respond to fast changing signals. However, reducing the time constant also reduces its gain which limits its sensitivity. As a result, the RC differentiator is not well suited for this application.

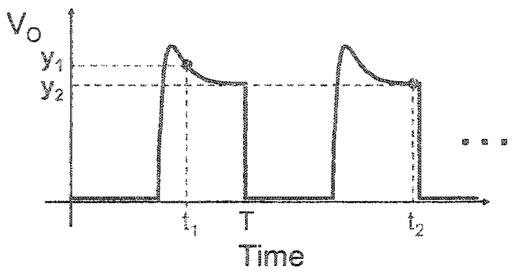


Fig. 11 Sampling scheme used to detect incomplete settling

An alternative approach that offers the potential to operate at much higher speeds is one that utilizes sampling. The waveforms shown in Fig. 11 illustrate the concept. The circuit is excited with a step input large enough to achieve a full-scale output swing. Fig. 11 shows two cycles of an output voltage swing from an amplifier that has a large pole-zero mismatch. An inter-period sample, y_1, is taken at time t_1 and compared to the settled value, y_2, at time T. If $y_2 - y_1 > 0$, it is deduced that the low-frequency pole lies to the left of the zero. Conversely, if $y_2 - y_1 < 0$, the pole lies to the right of the zero.

Obtaining the inter-period sample can disturb the subsequent settling, affecting the value of V_O at time T. Sampling the disturbed value at time T would introduce an error. To avoid this problem, the system is excited with a periodic step input and the settled-value y_2 is measured at the end of the subsequent cycle, at time $t = t_2$ ˜ $2T$.

Random noise present in the samples y_1 and y_2 limits the ability to resolve an underdamped system from an overdamped one. Averaging is used to combat the effects of random noise. By averaging the difference y1-y2 over k measurement cycles, the signal to noise ratio is improved by a factor of $\sqrt{k}$.

Performance Expectations

Because the step-response of a single-pole system never reaches its asymptotic steady-state value in finite time, adjusting the amplifier bias until the inter-period sample y_1 is equal to the end-of-period sample y_2 will not result in exact dipole cancellation. Therefore, even if the tuning circuit were perfect, there would be some residual mismatch due to an algorithmic bias. In reality, the tuning loop is imperfect itself limiting the performance even further. An understanding of how these factors affect the transient performance is required.

When used with negative feedback, the two-stage multipath compensated amplifier has a low-frequency dipole as illustrated in Fig. 9. Assuming linear operation, such a system's transient step response can be written as

$$y(t) = A_0 \cdot \xi \cdot \left[1 - (1+k) \cdot e^{P_1 \cdot t} + k \cdot e^{P_2 \cdot t}\right] \tag{10}$$

where A_0 and ξ are the closed-loop DC gain and the magnitude of the input-step respectively. The response is composed of a constant and two decaying exponential terms. The constant is equal to the asymptotic steady-state response, while the decaying exponentials appear due the system's finite ability to respond to fast-changing inputs. Since P_1 lies farther in the left half-plane than P_2, the $e^{P_1 \cdot t}$ term decays more quickly than the $e^{P_2 \cdot t}$ term. Thus, the $e^{P_2 \cdot t}$ term is often the bottleneck limiting system throughput. Note that the coefficient of the slow-settling term is k. The tuning scheme presented here aims to improve settling performance by reducing the magnitude of k to a level where the $e^{P_2 \cdot t}$ term does not adversely affect the settling time. Mathematically, k is given by

$$k = -\left(\frac{1 - P_2/Z}{1 - P_2/P_1}\right) \tag{11}$$

Note that k approaches zero as P_2 approaches Z. Thus, the magnitude of the slow-settling component can be reduced by increasing the accuracy of the pole-zero cancellation.

The tuning circuit adjusts the amplifier bias in an effort to make the inter-period sample y_1 equal to the end-of-period sample y_2. Since the tuning loop is imperfect, it is unable to make y_1 exactly equal to y_2. After tuning is complete, there is always some residual error. For the purpose of analysis, assume the tuning circuitry is capable of driving y_1 to within a certain tolerance (denoted by $\Delta/2$) of y_2.

$$(y_2 - \Delta/2) \leq y_1 \leq (y_2 + \Delta/2) \tag{12}$$

Substituting for y_1 and y_2 using (10) and solving for k at the endpoints of (12) yields two values of k

$$k_{1,2} = \frac{\left(e^{P_1 \cdot T_1} - e^{P_1 \cdot T_2}\right) \pm \frac{1}{2}\left(\frac{\Delta}{A_0 \cdot \xi}\right)}{\left(e^{P_2 \cdot T_1} - e^{P_2 \cdot T_2}\right) - \left(e^{P_1 \cdot T_1} - e^{P_1 \cdot T_2}\right)} \tag{13}$$

Assuming ξ is a full-scale input transition, then $A_0 \cdot \xi$ corresponds to a full-scale output transition. Therefore, since Δ is designates the magnitude of the tuning error, the term $\left(\frac{\Delta}{A_0 \cdot \xi}\right)$ can be thought of as the reciprocal of the effective resolution of the tuning loop. In accordance with common convention, resolution will be expressed in terms of *bits* of resolution. Thus, $\left(\frac{\Delta}{A_0 \cdot \xi}\right)$ will be replaced with 2^{-N} where N is the effective resolution of the tuning loop in bits.

Although the analytical expression of (13) is quite succinct, the relationship is difficult to visualize for two reasons. First, there are too many parameters that can be varied. Relationships that have more than 3 dimensions can be hard to visualize. Second, the relationship between the model parameters and k is nonlinear. Despite these problems, Fig. 12 provides useful insight into the relationship. It shows the worst-case $|k|$ as a function of the tuning loop resolution assuming the sampling instances T_1 and T_2 are chosen optimally. The plotted curves correspond to different pole ratios $\rho = P_2 / P_1$. Only stable systems that have a low-frequency dipole are of interest, therefore, ρ lies in the range $0 < \rho < 1$. The curves in the plot reveal how $|k|$ varies over the normal range of ρ. Each of the curves trend downward indicating that, in general, the magnitude of the slow-settling component declines as the resolution of the tuning loop increases. Additionally, for a given tuning loop resolution, the larger the pole ratio, the smaller the magnitude of k. Given these observations, it is apparent that systems that benefit the most from tuning by the described method are those with widely spaced poles. It should be noted that the curves of Fig. 12 represent the *best result* that you can expect to achieve for a system with a given tuning loop resolution and pole-ratio. If T_1 and T_2 are chosen at points that differ from the optimal, inferior results can be expected.

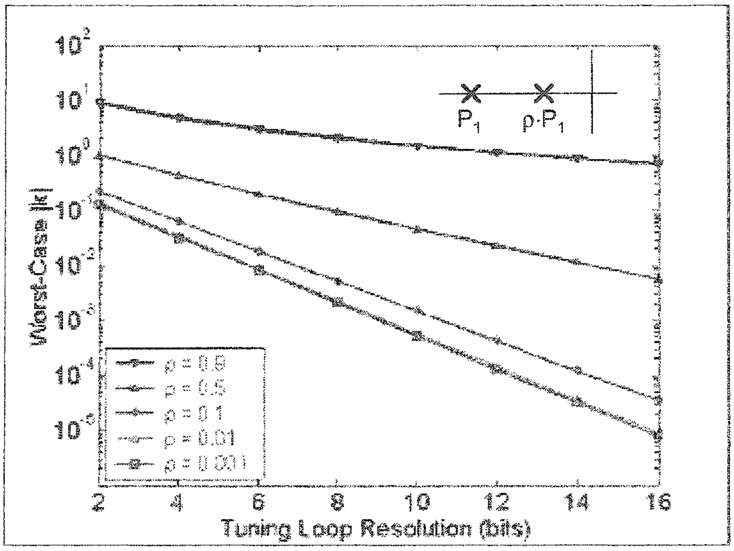


Fig. 12 Worst-case $|k|$ as function of tuning loop resolution assuming optimal choice of T_1 and T_2

Implementation

The calibration technique was implemented by the architecture shown in Fig. 13. The normal data path consists of the switched capacitor gain stage of Fig. 7 constructed with the two-stage multipath feedforward compensated amplifier of Fig. 8. The amplifier's second-stage bias, I_{DC} is fixed, while the first stage bias, I_{BIAS} is varied to adjust the pole and zero locations. Table 1 shows how the simulated pole and zero locations vary with respect to I_{BIAS}. Note that P_2 cancels Z_1 somewhere in the interval $0.45 < I_{BIAS} < 0.5$ μA. To demonstrate the low-voltage compatibility of the multipath feedforward compensation technique, the amplifier was designed to operate at 1.5V with a 1V differential output swing. For convenience, the remainder of the circuitry was designed to operate at 2.5V. The components in the shaded areas of Fig. 13 comprise the calibration loop which consists of two switched capacitor (SC) integrators, a crossbar switch, and a voltage-controlled current source.

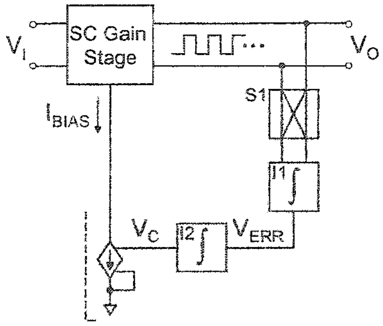


Fig. 13 Block diagram of system under calibration, shading identifies calibration circuitry

During normal operation, I_{BIAS} is held constant because V_C is held fixed by integrator I_2. Due to the existence of leakage currents, this configuration requires periodic refreshing to maintain the accuracy of the cancellation. For applications where periodic refresh is not acceptable, alternative architectures that generate the required control voltage using a DAC may be required.

Table 1 Simulated open-loop pole and zero locations as a function of IBIAS

		Open-Loop			Closed-Loop	
IBIAS	Z1	P1	P2	DC gain	P1	P2
(uA)	$(\mu s)^{-1}$	$(ms)^{-1}$	$(\mu s)^{-1}$		$(\mu s)^{-1}$	$(\mu s)^{-1}$
0.20	-12.9	-20.1	-28.3	1996	-63.2	-8.97
0.25	-15.7	-11.6	-26.7	4468	-58.9	-11.7
0.30	-18.1	-9.66	-26.3	6273	-56.1	-14.2
0.35	-20.2	-9.60	-26.0	7123	-53.4	-16.7
0.40	-22.1	-10.2	-25.8	7427	-50.6	-19.3
0.45	-24.0	-11.0	-25.6	7497	-47.5	-22.2
0.50	-25.7	-11.9	-25.5	7466	-43.7	-25.9
0.55	-27.3	-12.8	-25.4	7392	-36.8	-32.7
0.60	-28.8	-13.7	-25.3	7302	-34.7 - 8.16i	-34.7 + 8.16i
0.65	-30.3	-14.7	-25.3	7205	-34.7 -11.6i	-34.7 +11.6i
0.70	-31.7	-15.6	-25.2	7107	-34.6 -14.1i	-34.6 +14.1i
0.75	-33.0	-16.5	-25.1	7012	-34.6 -16.1i	-34.6 +16.1i
0.80	-34.3	-17.4	-25.1	6919	-34.6 -17.8i	-34.6 +17.8i
0.85	-35.5	-18.3	-25.0	6830	-34.5 -19.3i	-34.5 +19.3i
0.90	-36.7	-19.2	-24.9	6745	-34.5 -20.7i	-34.5 +20.7i
0.95	-37.8	-20.1	-24.9	6662	-34.5 -21.9i	-34.5 +21.9i
1.00	-38.9	-21.0	-24.8	6584	-34.4 -23.0i	-34.4 +23.0i

Calibration is performed at startup and at periodic intervals to track temperature variations. Applications that require continuous operation require the instantiation of two systems. One processes data while the other is calibrated. To maintain the required level of performance without suspending the data flow, their roles are periodically reversed.

The waveforms of Fig. 14 illustrate the operation of the calibration routine. The circuit is excited with a periodic step input large enough to achieve a full-scale output swing. The calibration begins by resetting the signal V_{ERR} shown in Fig. 13. Next, the inter-period sample of the transient output voltage, y_1, is sampled onto the input capacitors of integrator I_1 at time t_1. At the start of the next cycle, *t=T*, the stored charge is transferred to the hold capacitors of I_1 producing a change in the output voltage of I_1, ? V_{ERR}, that is proportional to y_1. Switch S_1 is toggled every cycle. Thus, the polarity of the signal applied to I_1 alternates every cycle. The end-of-period sample, y_2, is sampled onto the input capacitors of I_1 at time t_2 and momentarily, the stored charge is transferred to the hold capacitors of I_1. Since the polarity of the input has been reversed, ? V_{ERR} offsets the previous ? V_{ERR} by an amount proportional to y_2. After completion of the described sequence of operations, the residual voltage at the output of I_1 is proportional to the desired quantity which is the difference between the magnitudes of the inter-period and end-of-period samples, y_1-y_2.

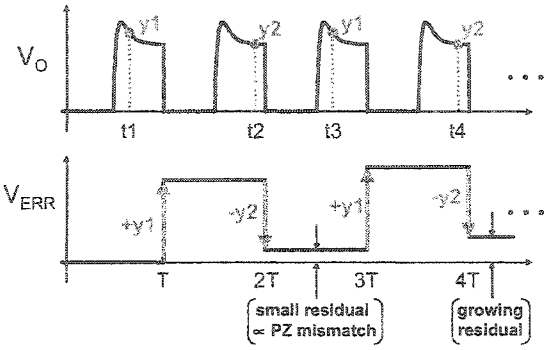


Fig. 14 Calibration loop waveforms

Repeating the $\{+y_1-y_2\}$ process without resetting V_{ERR} improves the system's ability to resolve the desired quantity from random sample noise. The desired signal adds linearly at the output of I_1 while the noise terms add as the square root of the sum of the squares. Therefore, the accumulation of V_{ERR} over N $\{+y_1-y_2\}$ cycles yields a $\sqrt{N}$ improvement in the signal to noise ratio.

If after N cycles of $\{+y_1-y_2\}$ accumulation, V_{ERR} is positive, it can be concluded that the zero leads the pole and I_{BIAS} must be reduced to obtain pole-zero cancellation. Conversely, if V_{ERR} is negative, the opposite holds true. Therefore, after V_{ERR} has been accumulated over N cycles, the second integrator, I_2, is strobed resulting in the slight adjustment of the control voltage, V_C, in the appropriate direction. The change in V_C modifies the bias current, I_{BIAS}, which in turn causes the poles to shift. The described procedure is repeated a fixed number of times and then the calibration process is terminated. The pseudocode of Fig. 15 summarizes the operation of the calibration routine. In the figure, N_C and N_I refer to integer constants that define the total number of times the pole locations are adjusted before termination and the number of $\{+y_1-y_2\}$ cycles to be accumulated before each V_C adjustment respectively. K_1 and K_2 are gain constants that are determined by capacitor ratios. They are chosen to be small to ensure that the steps taken by the calibration are small enough to avoid instability in the calibration algorithm.

```
$V_C = 0$;
for C=1:$N_C$ {
    $V_{ERR} = 0$;
    for I=1:$N_I$ {
        $V_{ERR} = V_{ERR} + K_1 \cdot (y_1 - y_2)$
    }
    $V_C = V_C + K_2 \cdot V_{ERR}$
}
```

Fig. 15 Pseudocode description of the calibration algorithm

Integrators I_1 and I_2 were realized with switched capacitor integrators of the type depicted in Fig. 16. To avoid static pole-zero mismatch, structures that include offset cancellation were chosen. The finite gain of the amplifiers limit the accuracy of the charge transfer operations and therefore limit the overall accuracy of the post-calibration pole-zero cancellation. Considering the finite gain effects but neglecting amplifier offset, the charge delivered per cycle to C_2, Q[n], for the integrator shown in Fig. 16 is given by:

$$Q[n] = (1-k)C_1V_{in}[n] - kC_2V_{out}[n-1] \tag{14}$$

where $k = 1/[C_1 + (A+1)C_2]$. For a large DC gain, A, k approaches zero and the charge transferred each cycle is the desired value of $C_1 \cdot V_{in}$. However, for smaller DC gains, k becomes nonzero. As a result, due to the second term, the amount of charge actually delivered is slightly affected by the previous output voltage held by the integrator.

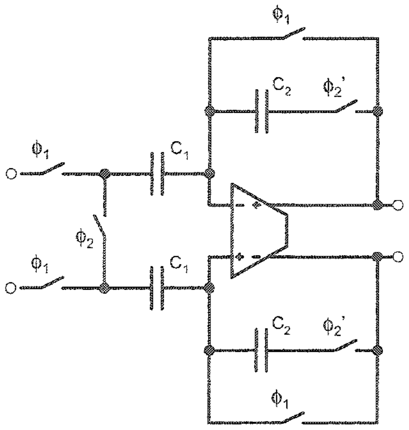


Fig. 16 Switched capacitor integrator used for I_1 and I_2

This is problematic because it leads to an undesired drift in the output voltage that is incorrectly interpreted as a pole-zero mismatch. Fig. 17(a) illustrates the output voltage of integrator I_1 for $\{+y_1-y_2\}$ phasing. As illustrated in the figure, the output voltages preceding the y_2 transitions are always positive and larger than the voltages preceding the y_1 transitions. Due to the second term in (14), the y_1 and y_2 related charge transfers are not exactly equal. Despite the fact that $y_1=y_2$, a systematic output drift occurs which is incorrectly interpreted as pole-zero mismatch.

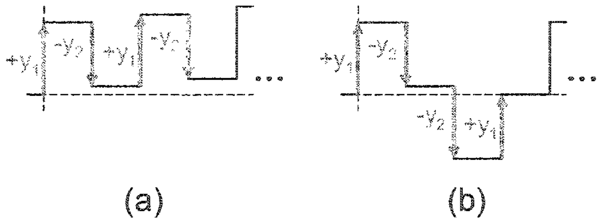


Fig. 17 Integrator output waveforms for (a) {+y1-y2} sequence and (b) {+y1-y2-y2-y1} sequence

Modifying the order of the charge add-subtract operations reduces the sensitivity to the finite gain of the amplifiers and diminishes the resultant output drift. Use the $\{+y_1-y_2-y_2+y_1\}$ sequence illustrated in Fig. 17(b) rather than the $\{+y_1-y_2\}$ sequence. Notice that the output voltages preceding one of the y_1 and one of the y_2 transitions are large in magnitude. Therefore, the errors due to the second term of (14) can be significant. However, due to the fact that the output voltages preceding those transitions differ in polarity, their effects partially offset each other and thus reduce the problem of output drift. The voltages preceding the other two transitions are generally small and therefore according to (14) result in smaller errors.

The most direct way to gauge the effectiveness of the calibration technique is to observe the settling behavior of the transient step response waveforms. Due to the fast transitions in the response, these signals have significant energy at high frequencies. Exporting the signal to an external instrument without compromising its spectral content requires a driver with a very wide bandwidth. To reduce the bandwidth requirement to the point where it became feasible to design the driver, it was necessary to slow the circuit down by adding additional capacitive loading. The resultant driver is shown in Fig. 18. It drives the 50-Ohm inputs of an oscilloscope and consumes 30mW of power.

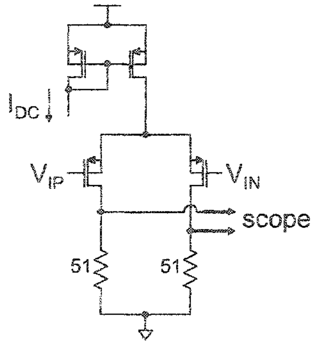


Fig. 18 Driver circuit

Measured Results

A prototype chip was fabricated in a 0.25μ CMOS process on a 2.2 x 2.2 mm die. A micrograph that has been annotated with labels identifying the major functional system blocks is shown in Fig. 19. The switched capacitor gain stage whose response is being tuned consumes roughly 0.08 mm^2 while the circuitry required to support the tuning procedure consumes nearly three times that area.

Fig. 20 shows the range over which the system's transient step response can be adjusted. The differential control voltage normally produced at the output of the tuning loop (V_C) was applied externally and swept over its normal range. The plot shows that the response is continuously adjustable between a significantly overdamped response to one that is obviously underdamped.

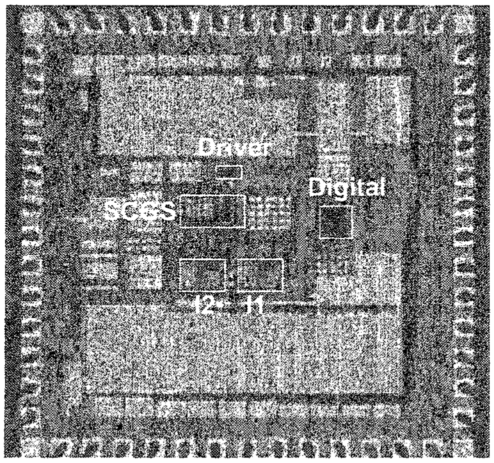


Fig. 19 Chip Micrograph

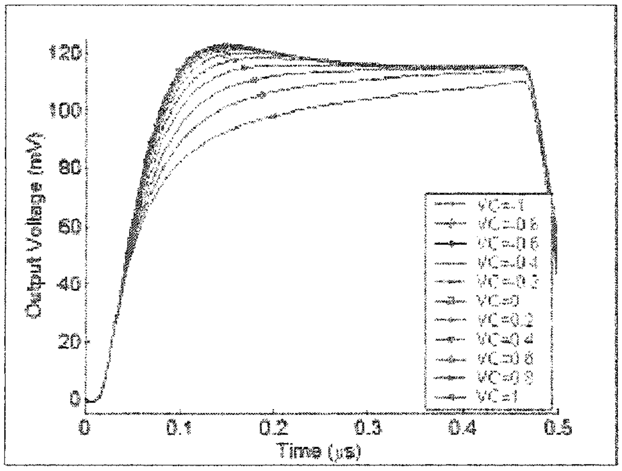
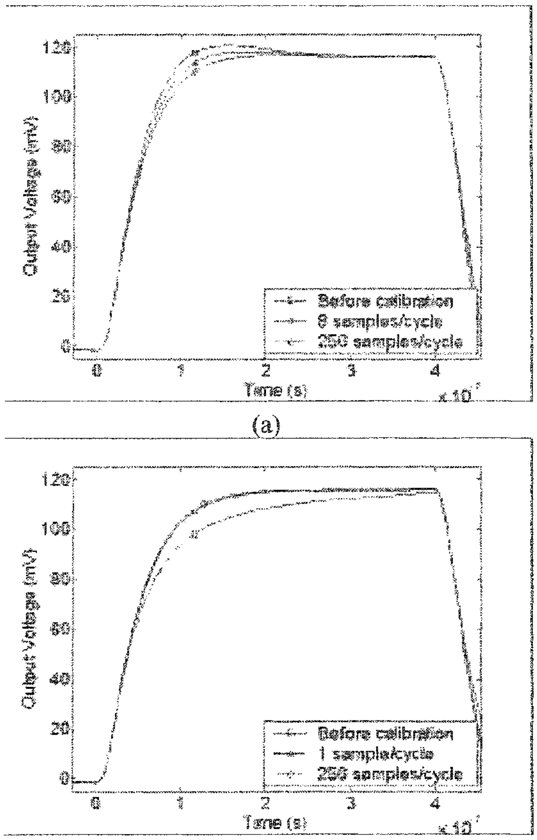


Fig. 20 Measured transient step response as a function of tuning loop control voltage V_C

In an effort to simulate the system's ability to compensate for process, aging, and environmental variations, the uncalibrated amplifier was intentionally biased to elicit a non-ideal response. Then, calibration was enabled and the system was allowed to tune itself to compensate for the bias mismatch. The results were recorded and are shown in Fig. 21.

Fig. 21(a) shows the case where the uncalibrated amplifier was biased to exhibit an underdamped response. Two different post-tuning responses are shown. One corresponds to accumulating V_{ERR} over 8 I_1 {+y1-y2-y2+y1} adjustment cycles while the other was obtained by accumulating over 256 cycles. As expected, as the number of accumulation cycles increases, the tuning loop resolution improves forcing y_1 closer to y_2.

Fig. 21(b) shows an overdamped case. Note the dramatic improvement in the response obtained by accumulating V_{ERR} over 1 cycle. Less dramatic effects are obtained by accumulating V_{ERR} over a larger number of cycles.



(a)

(b)

Fig. 21 Measured transient step responses before and after calibration (a) underdamped response (b) overdamped response

The simulated pole locations are listed in Table 1. When P_2 cancels Z_1, the closed-loop pole ratio P_1/P_2 is approximately 1.6. Fig. 22 is a plot that shows the simulated effect of low-frequency dipole mismatch on settling time for 1, 0.5, and 0.1% settling accuracies. The plot is only valid for systems with pole ratios of 1.6 and was obtained by fixing the pole ratio and sweeping the zero over the low-frequency pole [19]. The prototype does not conform to the curves of Fig. 22 exactly because the pole ratio of a real system does not remain constant as I_{BIAS} is adjusted. Even though the curves are inexact, they offer a reasonable approximation of the circuit's behavior and provide plenty of insight.

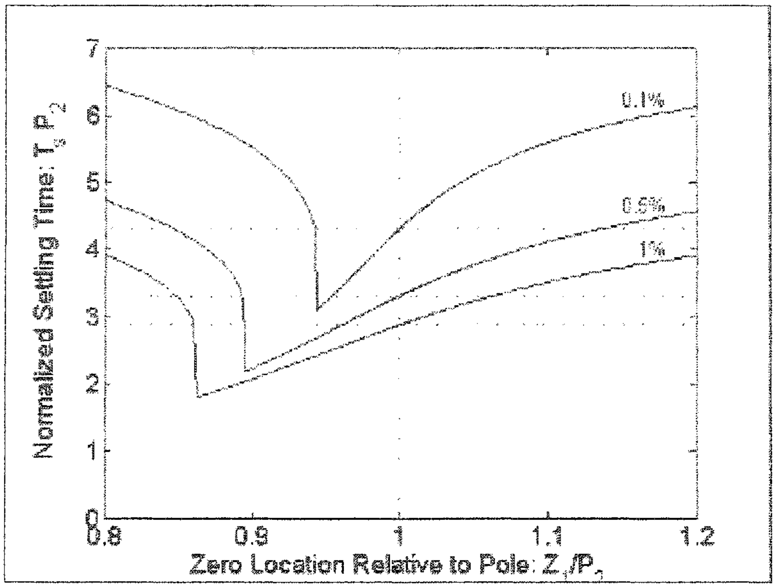
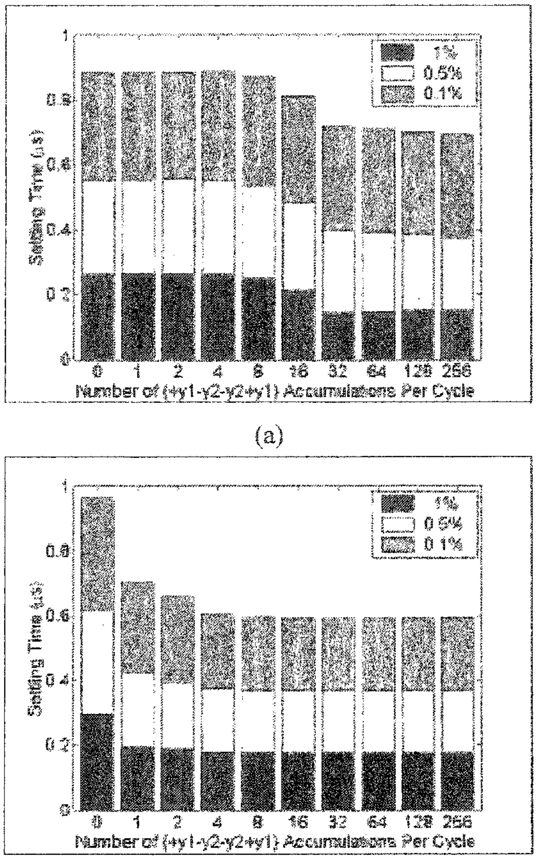


Fig. 22 Simulated settling time of a system with a low-frequency dipole and pole ratio $P_1/P_2 = 1.6$ vs. dipole mismatch for settling accuracies of 1, 0.5, and 0.1%

The vertical dotted line in Fig. 22 represents cases where the low-frequency dipole cancellation is exact. In those cases, settling time is determined solely by the high-frequency pole, P_1. Thus, the horizontal dotted lines represent the times required for a perfect system to settle to 1, 0.5, and 0.1% accuracies. Observe that settling time can actually be improved by introducing a slight pole-zero mismatch. By placing the zero slightly lower in frequency than P_2, the system will settle faster than a system with a perfectly cancelled dipole.

Overall, when the dipole mismatch is large, the settling time is primarily determined by the location of the low-frequency pole P_2. As the cancellation accuracy improves, settling is increasingly dependent upon the location of the high-frequency pole P_1.

Fig. 23 shows the measured settling time as a function of the number of {+y1-y2-y2+y1} cycles used to accumulate V_{ERR} for 1, 0.5, and 0.1% settling accuracies. Fig. 23(a) and (b) correspond to the under- and overdamped cases of Fig 21(a) and (b) respectively. The leftmost bars labeled '0' correspond to the systems before calibration.



(a)

(b)

Fig. 23 Measured settling time vs. number of {+y1-y2-y2-y1} cycles used to determine V_{ERR} for settling accuracies of 1, 0.5, and 0.1% (a) for underdamped response of Fig. 21(a), (b) for overdamped response of Fig. 21(b)

Increasing the number of {+y1-y2-y2+y1} accumulations (proceeding from left to right in Fig. 23(a)), results in increasing pole-zero cancellation accuracy and therefore, corresponds to moving from somewhere starting on the left side of Fig. 22 toward the middle of the figure ($Z_1/P_2 = 1$). Likewise, proceeding from left to right in Fig. 23(b) corresponds to starting somewhere on the right side of Fig. 22 and moving toward the middle of the figure.

Summary

Multipath feedforward compensated amplifiers are not suitable for applications that require fast settling because they have low frequency dipoles that result in the appearance of slow-settling components in the transient response. A self-calibration methodology that compresses the dipole mismatch was outlined for a two-stage structure. A prototype circuit was integrated in a 0.25μ CMOS process and the concept was experimentally verified.

Acknowledgment

Special thanks to David Cave and the entire DLTG group at Motorola for all of the assistance they provided. Especially Steve Jamison, Dave Doman, Hanyang Yang, Marshal Tiner, and Jim Rice. This work was supported in part by the R. J. Carver Charitable Trust.

References

[1] International Technology Roadmap for Semiconductors 2002 Update, Semiconductor Industry Association, San Jose, CA, 2002

[2] P. Yu, and H. S. Lee, "A High-Swing 2-V CMOS Operational Amplifier with Replica-Amp Gain Enhancement," *IEEE J. Solid-State Circuits*, vol. 28, pp. 1265-1272, Dec. 1993

[3] P. Yu, and H. S. Lee, "Settling Time Analysis of a Replica-Amp Gain Enhanced Operational Amplifier," *IEEE J. Solid-State Circuits*, vol. 42, pp. 137-142, Mar. 1995

[4] M. E. Schlarmann, S. Q. Malik, and R. L. Geiger, "Positive Feedback Gain-Enhancement Techniques for Amplifier Design," *in Proc. IEEE Int. Symp. on circuits and systems*, 2002

[5] G. C. Temes, "Autozeroing and Correlated Double Sampling Techniques," in Analog Circuit Design, J. H. Huijsing, R. J. van de Plassche, and W. M. C. Sansen, eds., Kluwer 1996

[6] S. Setty, and C. Toumazou, "Feedforward Compensation Techniques in the Design of Low Voltage Opamps and OTAs," in *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 1, pp. 464-467, 1998

[7] R. G. H. Eschauzier, and J. H. Huijsing, Frequency Compensation Techniques for Low-Power Operational Amplifiers, Boston : Kluwer Academic Publishers, 1995.

[8] E. M. Cherry, "Comments on A 100MHz 100 dB Operational Amplifier with Multipath Nested Miller Compensation Structure," *IEEE J. Solid-State Circuits*, vol. 31, pp. 753-754, May 1996

[9] E. M. Cherry, "A New Result in Negative Feedback Theory, and its Application to Audio Power Amplifiers," *Int. J. Circuit Theory*, vol. 6, pp. 265-288, July, 1978

[10] F. You, S. H. K. Embabi, and E. Sánchez-Sinencio, "Multistage Amplifier Topologies with Nested Gm-C Compensation," *IEEE J. Solid-State Circuits*, vol. 32, pp. 2000-2011, Dec. 1997

[11] K. N. Leung, P. K. T. Mok, W. Ki, and J. K. O. Sin, “Three-Stage Large Capacitive Load Amplifier with Damping-Factor-Control Frequency Compensation,” *IEEE J. Solid-State Circuits*, vol. 35, pp. 221-230, Feb. 2000

[12] K. N. Leung and P. K. T. Mok, “Analysis of Multistage Amplifier – Frequency Compensation,” *IEEE J. Solid-State Circuits*, vol. 48, pp. 1041-1056, Sep. 2001

[13] H. Lee and P. K. T. Mok, “Active-Feedback Frequency-Compensation Technique for Low-Power Multistage Amplifiers,” *IEEE J. Solid-State Circuits*, vol. 38, pp. 511-520, Mar. 2003

[14] J. Ng, R. M. Ziazadeh, and D. J. Allstot “A Multistage Amplifier Technique with Embedded Frequency Compensation,” *IEEE J. Solid-State Circuits*, vol. 34, pp. 339-347, Mar. 1999

[15] A. Thompson, E. de Angel, S. X. Wu, A. Amar, L. Wang, and W. Lee, “A DC Measurement IC with 130 nVPP Noise in 10Hz,” ISSCC Digest of Technical Papers, Feb. 2000

[16] A. Thomsen, D. Kasha, and W. Lee, “A Five Stage Chopper Stabilized Instrumentation Amplifier Using Feedforward Compensation,“ in 1998 Symp. VLSI Circuits, Digest of Technical Papers, 1998, pp. 220- 223

[17] M. E. Schlarmann, and R. L. Geiger, "A New Multipath Amplifier Design Technique for Enhancing Gain Without Sacrificing Bandwidth," in Proc. ISCAS, Vol. 2, Orlando, FL, May 30-June 2, 1999, pp. 610-615

[18] B. K. Thandri and J. Silva-Martinez, “A Robust Feedforward Compensation Scheme for Multistage Operational Transconductance Amplifiers with No Miller Capacitors,” *IEEE J. Solid-State Circuits*, vol. 38, pp. 237-243, Feb. 2003

[19] M. E. Schlarmann and R. L. Geiger, "Relationship Between Amplifier Settling Time and Pole-Zero Placements for Second-Order Systems," *Proc. MWSCAS 2000*, Lansing, MI, Aug. 8-11, 2000

[20] F. D. Waldhauer, "Analog Integrated Circuits of Large Bandwidth," in 1963 IEEE Conv. Rec., Part 2, pp. 200-207

[21] P. R. Gray, and R. G. Meyer, "Recent Advances in Monolithic Operational Amplifier Design," IEEE Trans. Circuits Syst., vol. CAS-21, pp. 317-327, May 1974

[22] B. Y. Kamath, R. G. Meyer, and P. R. Gray, “Relationship Between Frequency Response and Settling Time of Operational Amplifiers,” *IEEE J. Solid-State Circuits*, vol. SC-9, no. 6, pp. 347–352, 1974.

[23] M. E. Schlarmann and R. L. Geiger, "Use of the Newton-Raphson Iteration to Eliminate Low Frequency Dipoles," *Proc. MWSCAS 2002*, Tulsa, OK, Aug. 4-7, 2002

Appendix A. Design of a three-stage multipath feedforward amplifier

By nesting the basic multistage feedforward module of Fig. 4 within itself, amplifiers with more than two stages can be created. The procedure is illustrated for a three-stage voltage amplifier in Fig. 5. Two simple first-order gain stages, $H_1(s)$ and $H_2(s)$, are used to realize the basic module as shown in Fig. 5(a). Since only the notation has changed, all of the expressions derived for the basic module, equations (2)-(8), apply to this case except the 'a' and 'b' subscripts are changed to '1' and '2' respectively. Adjusting the bandwidth of the second stage facilitates pole-zero cancellation as described in (4). Therefore, the condition required for pole-zero cancellation is given by:

$$P_2 = P_1(1+A_1) = P_1 + GBW_1 \tag{15}$$

The partial transfer function is given by:

$$H_{12}(s) = \frac{V_2(s)}{V_i(s)} = \frac{A_2(1+A_1)}{1-s/P_1} \tag{16}$$

While the DC-gain, pole location and GBW are given by:

$$A_{12} = A_2(1+A_1) \tag{17}$$

$$P_{12} = P_1 \tag{18}$$

$$GBW_{12} = GBW_2 \tag{19}$$

Since the resultant structure of Fig. 5(a) has a first-order transfer function, it can be thought of as equivalent to a single stage and can be nested within another module as shown in Fig. 5(b). Since we are still dealing with the same module, equations (2)-(8) still apply except the 'a' and 'b' subscripts are replaced with '12' and '3' respectively. Thus, the requirements for pole-zero cancellation for this module are given by:

$$P_3 = P_{12}(1+A_{12}) = P_1 + GBW_{12} = P_1 + GBW_2 \tag{20}$$

When (20) is satisfied, the transfer function of the module in Fig. 5(b) is given by:

$$H_{123}(s) = \frac{V_3(s)}{V_i(s)} = \frac{A_3(1+A_{12})}{1-s/P_{12}} = \frac{A_3(1+A_2(1+A_1))}{1-s/P_1} \tag{21}$$

Expressions for the DC-gain, pole location and GBW are given by:

$$A_{123} = A_3\left(1 + A_2\left(1 + A_1\right)\right) \tag{22}$$

$$P_{123} = P_{12} = P_1 \tag{23}$$

$$GBW_{123} = GBW_3 \tag{24}$$

The resultant amplifier is shown in Fig. 5(c). Amplifiers with more than three stages can be constructed by additional nesting.

CHAPTER 3. USE OF THE NEWTON-RAPHSON ITERATION TO ELIMINATE LOW FREQUENCY DIPOLES

A paper published in the Proceedings of the

2002 Midwest Symposium on Circuits and Systems

Mark E. Schlarmann and Randall L. Geiger

Abstract

Amplifiers with closely-spaced low-frequency pole-zero pairs (dipoles) are normally avoided for applications that require fast settling because they have slow-settling components in the transient response. In this work, an algorithm that involves a Newton-Raphson iteration is utilized to tune an amplifier with multiple low-frequency dipoles and facilitate pole-zero cancellation. The tuned structure is more suitable for fast settling applications.

Introduction

As fabrication technology progresses into deep-submicron feature sizes, achieving an adequate DC gain is becoming increasingly difficult. The origin of this problem is twofold. First, reductions in supply voltages are making it difficult to employ cascoding and still maintain adequate signal swings. Second, the degradation in device output conductance is making it difficult to achieve an adequate gain in two or fewer non-cascoded stages. As a result, non-traditional amplifier topologies are being investigated with increased urgency.

Amplifiers with more than two stages of gain are a potential solution to the problem. However, to ensure their stability with negative feedback, multistage amplifiers need to be *compensated.* Since each additional stage introduces poles into the system transfer function, the task of compensation becomes more difficult as the number of stages is increased. Several multistage amplifier compensation strategies have appeared in the literature [24-27]. Unfortunately, as a side-effect of the compensation process, most techniques sacrifice the gain-bandwidth product of the amplifier in exchange for stability. As a result, amplifiers with three or more stages are typically too slow for applications that require fast settling.

There is at least one multistage amplifier compensation technique [28,29] that does not sacrifice the gain bandwidth product of an amplifier in exchange for achieving stability. Although the technique results in an amplifier that has a gain-bandwidth product that is as large as can be achieved with a single stage amplifier, the resultant structure is still not suitable for fast settling applications because it has a poor transient response. The limitation of this technique is the fact that it relies on the cancellation of low-frequency pole-zero pairs. Inexact cancellations result in the appearance of slow-settling components in the transient response [30,31] making these amplifiers unsuitable for applications that require fast accurate settling.

In an effort to overcome these limitations and extend the applicability of multistage multipath compensated amplifiers to the high-speed realm, a calibration technique to eliminate the dipole mismatch of a two-stage structure was proposed [32]. The viability of the technique was demonstrated in a 0.25μ CMOS process. The results are awaiting publication elsewhere.

The applicability of the method proposed in [32] is limited to amplifiers with two gain stages and one low-frequency dipole. In this paper we present a generalization of the technique to cover amplifiers composed of an arbitrary number of stages. Although we focus specifically on the multistage multipath compensated amplifier architecture proposed in [28], the technique is generally applicable to other architectures that suffer from low-frequency dipoles as well.

The problem and the assumptions required for its solution are briefly described in section 2. The new calibration technique is outlined in section 3 and an example and short discussion appear in section 4.

Problem Description and Assumptions

An *n*-stage multipath-compensated amplifier has a system transfer function that has *n* poles and *n-1* zeros. The details can be found in [28,29,32]. If the components of the system are prudently dimensioned, the zeros can be used to cancel all but one of the poles.

Fig. 24 shows a typical example of the closed-loop pole and zero locations in the complex *s*-plane for an *n*'th order multipath amplifier used in a standard feedback configuration. For notational purposes, the poles are numbered in the order of increasing magnitude. If perfect cancellation were possible, the system would be exactly first-order. Practically, mismatch always exists and the system only approximates a first-order response.

Imperfect cancellation of the pole-zero pairs (dipoles) results in the appearance of extra decaying exponential components in the transient response. These additional components will decay more slowly than the desired component because they lie at lower frequencies than the *uncovered* pole.

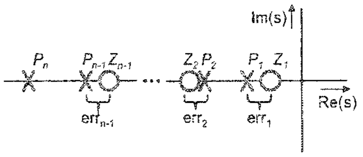


Fig. 24 Closed-loop pole and zero locations for an *n*'th order multipath amplifier

In this work it was assumed that the amplifier architecture allows the pole locations to be individually tuned. Fig. 25 illustrates the concept. The amplifier has a transfer function *H(s)* whose pole locations are adjustable via several control signals. The *k*'th control signal, b_k, is assumed to control the location of pole, p_k.

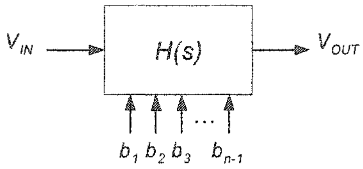


Fig. 25 Amplifier block with programmable pole locations

In a real physical implementation the *k*'th control signal is a voltage or current used to bias the *k*'th stage of the amplifier. In general, the relationship between a control signal and the corresponding pole location can be highly nonlinear. To ensure a reasonable model of the amplifier, a nonlinear relationship between the control signal and its corresponding pole location was assumed. The assumed relationship is shown Fig. 26. It is a hyperbolic tangent relationship scaled to allow tuning of ±25% of the pole's nominal value.

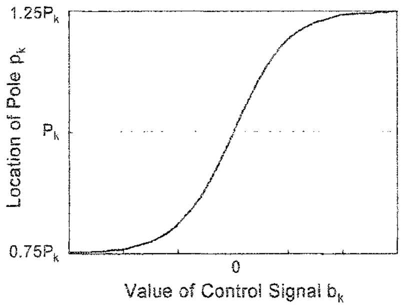


Fig. 26 Nonlinear relationship assumed between the control signals and their associated pole locations

Throughout the work we assume that the closed-loop poles of the system transfer function are widely separated on the real axis in the left half-plane. We also assume that each low frequency pole is located in close proximity (±15% their nominal values) to a zero. Although, the open-loop pole locations are highly sensitive to variations, feedback desensitizes these quantities and stabilizes their values. Therefore, these are reasonable assumptions for the *closed-loop* pole locations.

Proposed Calibration Technique

An n'th order system with a pole-zero map like the one shown in Fig. 24 has a transient step response given by:

$$y(t) = A_0\left(1 + \sum_{i=1}^{n} k_i \exp(p_i t)\right) \tag{25}$$

where A_0 is the asymptotic steady-state gain, p_i is the location of the i'th pole, and k_i is a constant defined by:

$$k_i = -\frac{\prod_{j=1}^{n-1}\left(\frac{p_i}{z_j} - 1\right)}{\prod_{\substack{j=1\\ j\neq i}}^{n}\left(\frac{p_i}{p_j} - 1\right)}, \qquad i = 1,2,\cdots n \tag{26}$$

The first term in (25) is the asymptotic steady-state response. The remaining terms all decay with time and form the transient component of the response. Thus the transient step response of an n'th order system with widely separated real left half-plane poles can be decomposed into a sum of n decaying exponentials with differing time-constants. This relationship is illustrated for a third-order system in Fig. 27. Observe that the total response shown in Fig. 27(e) is simply the sum of the components shown in Figs. 27(a)-(d).

From (26) you can see that that adjusting the i'th control signal b_i such that p_i and z_i are coincident forces k_i to zero. Thus, by careful adjustment of the n-1 control signals, the corresponding slow-settling terms in the transient response can be eliminated.

The calibration technique involves forcing the derivative of the transient step response to zero at specific instances in time. Judiciously choosing the points in time where the derivative is nulled ensures that each of the unwanted transient terms is eliminated.

The slowest settling component of the transient response is the term associated with p_1. Since the poles are widely separated, the other transient components decay significantly faster. Forcing the derivative of the step response to zero at a point in time after all the other transient components have decayed away ensures that the slowest settling component's coefficient, k_1, is forced to zero.

The next slowest settling component of the transient response is associated with p_2. Since the poles are widely separated, the other transient components, disregarding the one associated with p_1, decay significantly faster.

Forcing the derivative of the step response to zero at a point in time, T_2, after all the other transient components, except the p_1 term, have decayed away ensures that the joint variation due to the action of both the p_1 and p_2 terms sums to zero. However, since we forced the p_1 term to zero by choosing a time point at T_1, the joint effect of zeroing the derivative at times T_1 and T_2 is that the p_2 term must also be forced to zero.

The same arguments can be used justify repeated application of the concept until all n-1 slow-settling components are eliminated.

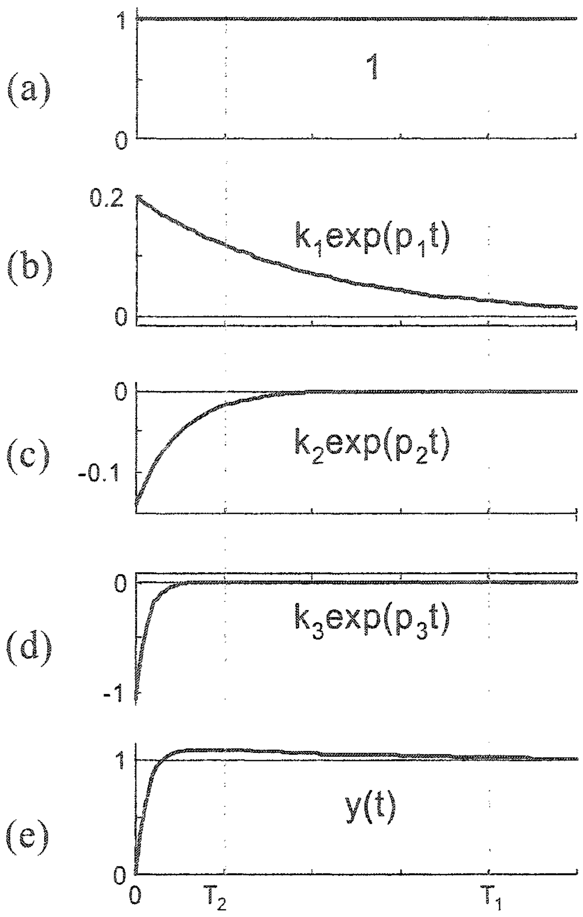


Fig. 27 Total transient step response of a third-order system (e) can be decomposed into a summation of (a) thru (d)

For example, forcing the derivative of the step response shown in Fig. 27 to zero at time T_1 requires that $k_1 \approx 0$. Simultaneously requiring that the derivative of the response is zero at time T_2 ensures that $k_2 \approx 0$ as well.

Reasonably good performance is obtained by choosing the time points equal to two time constants.

$$T_i = 2\tau_i = -\frac{2}{p_i}, \qquad i = 1,2,\cdots n-1 \tag{27}$$

In mathematical terms, minimization of the derivatives at the suggested time points can be written as:

$$f_1(b_1, b_2 \cdots b_{n-1}) = \left.\frac{\partial y}{\partial t}\right|_{t=T_1} = 0 \tag{28}$$

$$f_2(b_1, b_2 \cdots b_{n-1}) = \left.\frac{\partial y}{\partial t}\right|_{t=T_2} = 0 \tag{29}$$

$$\vdots$$

$$f_{n-1}(b_1, b_2 \cdots b_{n-1}) = \left.\frac{\partial y}{\partial t}\right|_{t=T_{n-1}} = 0 \tag{30}$$

where the f_i's are nonlinear functions of the control signals. In vector notation, these equations can be written as:

$$\mathbf{f}(\mathbf{b}) = \mathbf{0} \tag{31}$$

Equation (31) is a system of *n-1* nonlinear equations in *n-1* unknowns. The Newton-Raphson algorithm is one technique that is commonly used to solve these types of problem and is described elsewhere [33,34]. One drawback of this approach is that the derivatives of each of the nonlinear functions with respect to each of the control signals is required at each iteration. The required derivatives can be obtained using a finite difference method on samples of the transient *step* response. An easier method to obtain the same information involves sampling the *impulse* response.

Example

A fourth-order linear system with poles spaced at an interval of a decade was assumed. Using a normal distribution with a standard deviation equal to 15% of the magnitude of the associated pole, three zeros were randomly generated near the low-frequency poles.

Nonlinear relationships similar to the one shown in Fig. 26 were assumed to relate the control signals to the pole locations.

Table 2 shows the locations of the poles and zeros prior to and after calibration. The algorithm converged in 6 iterations. Note that there were significant dipole mismatches prior to calibration, but after calibration, the results agree to better than 6 significant digits.

Table 2 Pole and zero locations before and after the calibration routine was performed.

	Poles (pre-cal)	Zeros	Poles (post-cal)	
	-1e3	-1.134345e3	-1.134345e3	
	-1e4	-1.109643e4	-1.109643e4	
	-1e5	-1.086678e5	-1.086678e5	
	-1e6		-1.000000e6	

The algorithm has been used to successfully tune systems as large as 9'th order containing 8 dipoles. The technique has also been used to tune structures with poles spaced as close as an octave apart. Convergence problems may arise if the magnitude of the pole-zero mismatch is on the same order as the spacing between the poles.

Before a practical implementation of this algorithm can be implemented, the effects of noise and quantization need to be considered.

Summary

Amplifiers with low-frequency pole-zero pairs are not suitable for applications that require fast, accurate settling because dipole mismatches result in slow settling components in the transient response. In an effort to overcome this limitation, a technique to tune the responses of amplifiers to eliminate the dipole mismatch is proposed.

The procedure requires sampling the transient step response and uses the Newton-Raphson algorithm to determine the values of the bias currents and voltages required to achieve accurate cancellations.

To demonstrate the technique a 4'th order system with 3 mismatched low-frequency dipoles was calibrated. After six iterations, the dipoles matched to better than 6 significant digits.

Acknowledgment

This work was supported, in part, by the Semiconductor Research Corporation and National Science Foundation.

References

[24] R. G. H. Eschauzier, and J. H. Huijsing, Frequency Compensation Techniques for Low-Power Operational Amplifiers, Boston : Kluwer Academic Publishers, 1995.

[25] F. You, S. H. K. Embabi, and E. Sánchez-Sinencio, "Multistage Amplifier Topologies with Nested Gm-C Compensation," *IEEE J. Solid-State Circuits*, vol. 32, pp. 2000-2011, Dec. 1997

[26] K. N. Leung and P. K. T. Mok, "Analysis of Multistage Amplifier – Frequency Compensation," *IEEE J. Solid-State Circuits*, vol. 48, pp. 1041-1056, Sep. 2001

[27] J. Ng, R. M. Ziazadeh, and D. J. Allstot “A Multistage Amplifier Technique with Embedded Frequency Compensation,” *IEEE J. Solid-State Circuits*, vol. 34, pp. 339-347, Mar. 1999

[28] M. E. Schlarmann, and R. L. Geiger, "A New Multipath Amplifier Design Technique for Enhancing Gain Without Sacrificing Bandwidth," *in Proc. ISCAS*, Vol. 2, Orlando, FL, May 30-June 2, 1999, pp. 610-615

[29] B. Kumar Thandri S, J. Silva-Martinez, and F. Maloberti, “A Feedforward Compensation Scheme for High Gain Wideband Amplifiers,” in Proc. ICECS 2001, Vol. 3, 2001, pp. 1115-1118 vol.3

[30] F. D. Waldhauer, "Analog Integrated Circuits of Large Bandwidth," in 1963 IEEE Conv. Rec., Part 2, pp. 200-207

[31] B. Y. Kamath, R. G. Meyer, and P. R. Gray, “Relationship Between Frequency Response and Settling Time of Operational Amplifiers,” *IEEE J. Solid-State Circuits*, vol. SC-9, no. 6, pp. 347–352, 1974.

[32] M. E. Schlarmann, and R. L. Geiger, "Technique to Eliminate Slow-Settling Components that Appear Due to Dipoles," *in Proc. MWSCAS 2001*, Vol. 1, Dayton, OH, Aug 14-17, 2001, pp. 74-77

[33] D. Kahaner, C. Moler, and S. Nash, Numerical Methods and Software, Englewood Cliffs, NJ: Prentice Hall, 1989

[34] J. Vlach and K. Singhal, Computer Methods for Circuit Analysis and Design, New York: Van Nostrand Reinhold, 2nd Ed., 1994

CHAPTER 4. DESIGN SPACE EXPLORER: A WORLD-WIDE-WEB BASED ANALOG CIRCUIT DESIGN TOOL AND DESIGN KNOWLEDGE REPOSITORY

A paper to be submitted to an IEEE Journal

Mark E. Schlarmann and Randall L. Geiger

Abstract

A prototype network-centric circuit design tool and design knowledge repository has been developed. They allow a designer to interactively explore a circuit design space of pre-characterized circuit topologies using a convenient graphical user interface. Users can extend the system to include new or custom circuit topologies without a lot of programming effort by writing their own design specification files.

Designers will benefit from the use of the application as it will enable them to obtain a deeper understanding of the operation of their circuits and the performance tradeoffs that are possible for a given circuit topology. The enhanced understanding gained by the designers will result in better design realizations and will accelerate the development of improved topologies in the future.

Several benefits accrue due to the use of the network-centric computing paradigm of which the most significant is the enhanced communication of design knowledge and prevention of reinvention by the use of a centralized *Design Knowledge Repository*.

Introduction

Digital integrated circuits are routinely designed, synthesized, placed and routed, and verified using highly optimized computer-aided design tools. Considerable efforts have been expended in academia [35-38] and industry [40-42] in attempts to apply similar design automation concepts to analog systems. However, almost all commercial analog circuits are still designed by hand. The limited success of analog design automation is often attributed to the complexity of the analog design problem itself. Analog design is a very knowledge-intensive task that is presently best performed by humans.

Most of the design automation systems reported in the literature complete the entire design process with little or no human interaction. They are often criticized for their complicated nature and steep learning curves. Due to the complex nature of the general analog design problem, instead of attempting to solve the entire design problem in one automated step, it may be advantageous to not completely remove human interaction from the process at the outset. Rather, more practical benefit may be derived from small, easily used, interactive design tools that analog designers can use to make their jobs easier. With this approach, each utility will be used for only a small portion of the overall design task and humans will still guide the process.

Accelerated Assimilation of Designer Knowledge

The standard technique that is commonly taught for solving a multidimensional circuit design problem is to define an application specific cost-function and subsequently use an optimization routine to determine the design parameters that minimize it. Practically, circuits are seldom designed using this methodology because writing a cost-function that realistically represents a designer's priorities is a complicated and time-consuming task that depends upon the application under consideration. Attempting to short-circuit or approximate the cost function results in circuit designs that are sub-optimal.

As a side effect, the use of mathematical optimization tools does not substantially contribute to the designer's understanding of the relationships among the performance parameters and the design parameters. This is problematic because coming up with improved circuit designs and circuit topologies is a creative process that draws upon a designer's experience and understanding. The use of automated optimization procedures may actually impede the development of improved circuit topologies because the non-interactive nature of an automated optimization approach results in slower assimilation of designer knowledge.

Designers can more rapidly deepen their understanding of a design by interactively exploring a design space and observing the corresponding effects on and the relationships between the performance parameters. The benefits are especially clear for designs that are too complicated to derive tractable analytical relationships for the performance parameters. The additional insight into design performance gained by manual exploration of the design space will aid the designer in creating improved design topologies in the future.

Duplication of Effort

Reinvention is another problem that plagues the analog circuit design community. Many of the same circuit problems arise over and over again in slightly different contexts. In most cases, the resultant solutions are not sufficiently generalized to be easily applied to future applications, and if they are, there is often not a practical mechanism to communicate the results to the larger design community. Even groups within the same organization commonly duplicate each other's efforts.

Aggregation of generalized design knowledge in one central location that is accessible to everyone within an organization through a standardized interface could result in substantial productivity gains.

The Prototype Application

To address the problems previously discussed, a prototype application consisting of a network-centric circuit design tool and a design knowledge repository have been developed.

Although the software can also be run as a stand-alone application, the prototype system conforms to the emerging network-centric computing paradigm. Fig. 28 contains a block diagram illustrating the network-centric operation of the tool. The executable, in the form of a Java applet, is transmitted to any Java capable web browser via a network connection. A repository of pre-defined design specification files is maintained at the server. Each of the design specification files embodies the design knowledge available for a specific circuit topology that can be downloaded and used to program the applet at the request of the client.

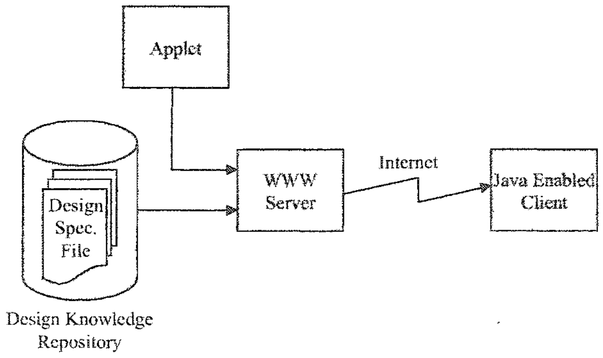


Fig. 28 Block diagram illustrating network-centric operation

The downloadable applet is used to interactively explore a design space through a graphical user interface. The required design information is defined by loading a text-based design specification file that contains a list of equations used to characterize the behavior of the circuit and a list of constraints that restrict the solution to the domain of viable solutions.

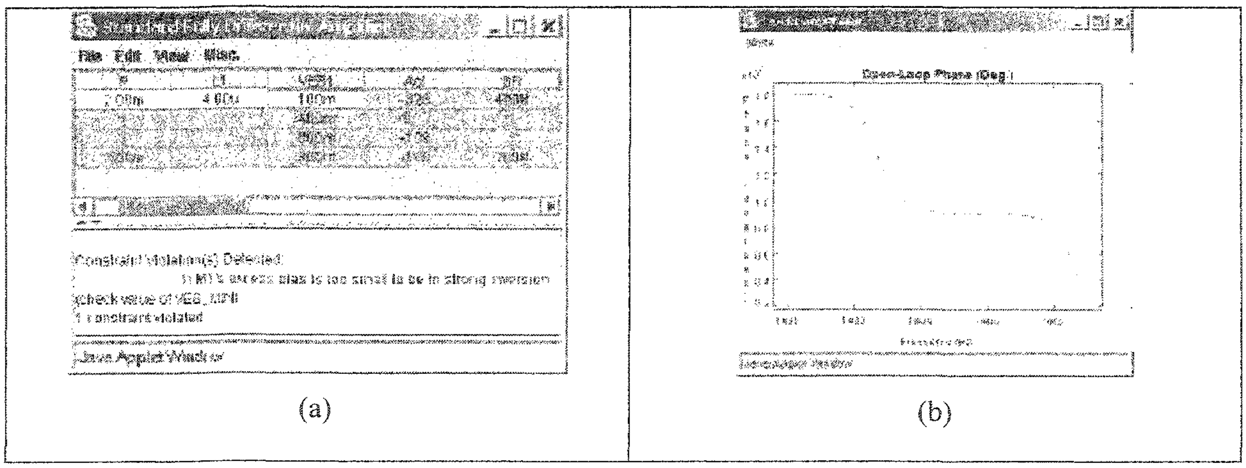
The application is similar in concept to the equation manipulator developed by Swings, Gielen, and Sansen called DONALD in [38]. As described in their publication, this tool could eventually become part of a larger overall automated analog design system. Our realization differs from theirs in that the system architecture was chosen to be network-centric and the equations are required to be declared in imperative form.

The application's interactive nature will allow designers to more quickly obtain a deeper understanding of how their circuits operate and what performance tradeoffs are possible for a given circuit topology. The additional insight gained through the use of the tool will accelerate the development of improved circuit topologies in the future. Reinvention is avoided because the design specification file for a given circuit topology is developed once by one author and then "published" for others to use.

Downloadable design specification files can be quickly used to investigate the characteristics of certain pre-characterized circuit topologies. Any user can extend the system to include new circuit topologies by developing their own custom design specification files that upon completion can be published for others to use by submission to a centralized design knowledge repository.

Fig. 29(a) contains an example screen snapshot of a Design Space Explorer application window. The window is divided horizontally into two resizable panes. The lower pane, called the *feedback pane*, contains a text area used for providing interactive feedback to the user while the upper pane contains a table of design parameter data.

The first row in the table is a special row for editing the values of the design parameters. It is referred to as the *current values row*. The numerical values of the parameters in the first row are referred to as the *current values* of the parameters. The cells with white backgrounds are modifiable while those with gray backgrounds are either declared as constants or computed and not modifiable. The remaining table rows each correspond to different viable designs.



Constraint Violation(s) Detected:
1) M1's excess bias is too small to be in strong inversion
check value of VEG_MIN
1 constraint violated

Java Applet Window

(a)

(b)

Fig. 29 Screen snapshots (a) Design Space Explorer Application Window, (b) an example plot

Figure 30 shows the tool's operational flowchart. Most of the time the application is idle awaiting new input data. When a user enters new data, the tool computes all of the dependent parameters and then checks to see if the solution violates any of the constraints. If any constraints are violated, an error message is generated and fed back to the user via the feedback pane. However, if none of the constraints were violated, the design is considered *viable* and an entry corresponding to the design is added to the table of viable designs. In this manner, the application automatically maintains a history of all of the viable designs encountered. This is a valuable feature for designers because they can review and compare the effects of the various parameter adjustments throughout their session. Upon successful addition of a design to the table of viable designs, an informational message is generated and displayed in the feedback pane.

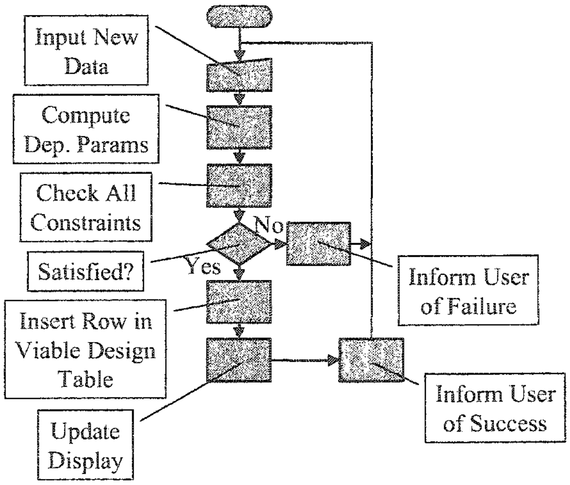


Fig. 30 Operational flowchart

In the screen snapshot of Fig. 29, three viable designs were present in the table in rows 2 through 4 at the time the snapshot was taken. Each of the three viable designs satisfies all of the applicable constraints but since the independent parameters for each of the three designs differ, they have slightly different values of the performance parameters. As indicated in the feedback pane, the *current design* in row 1 is not viable because it does not satisfy all of the design constraints.

The interface has been designed to enhance productivity. The user is able to control parameter visibility and display order in the table of viable designs. Plotting capability has been integrated to rapidly convey the meaning of complex functional relationships. Fig. 29(b) shows a typical example of a plot created by Design Space Explorer. Special rendering techniques involving special characters and coloring are employed to help users quickly identify the differences between the current design and other designs in the table. These features help focus the user's attention on the parameters of interest, shielding them from unnecessary complexity. Additionally, the user is given the ability to reorder the rows of the table of viable designs based upon a column-wise sorting criterion. This allows the user to quickly rank the viable designs based upon the parameters of interest.

Design Specification Files

In architecting the Design Space Explorer application, every attempt was made to keep the main application general. By making the application programmable, its use was not restricted to a specific discipline. Design Space Explorer is equally able to model, for example, mechanical and thermodynamic systems as it is able to model electrical systems. For the work presented here, however, the focus is on modeling electrical systems.

To facilitate user customization of existing design specifications and encourage the development of new design specifications, *Design Space Explorer* accepts plain-text design specification input files. Presently, these files are hand-coded using a standard text editor. In the future, a customized editor will be developed to ease the burden of creating these files.

The input file language grammar is very similar to the extensible markup language (XML) specification. The file is simply a list of *elements* that are denoted by *tags*. Although there are many different elements in the design specification file grammer, the bulk of a design specification file is made up of *equation* and *constraint* elements.

Equation elements define symbolic variables via a mathematical expression involving zero or more other variables. In a design specification file, these equations model the response of the system. In order to function correctly, the dependency graph defined by the relationships among the equations must be *acyclic*. In the future, we hope relax this requirement by incorporating a nonlinear equation solver.

Constraint elements define mathematical inequalities that are used to distinguish viable designs from those that are not viable. Examples of typical constraints include *feasibility constraints* such as range-restrictions on signals, or *performance constraints* like minimum or maximum values for specified performance parameters. Flagging those designs that do not satisfy the all of constraints is important to prevent the user from drawing false conclusions.

The following list outlines the procedure usually employed to develop a design specification file.

1) Identify the design degrees of freedom (*independent parameters*).

2) Express the performance parameters (*dependent parameters*) in terms of the independent parameters and other dependent parameters with the provision that no dependency rings are created.

3) Express all solution constraints in terms of the independent and dependent parameters.

4) Assemble the results from steps (1), (2) and (3) into an input file of the specified syntax.

Simplified Example: A common-source amplifier

This section provides a simplified example to illustrate the procedure of writing a design specification file. In the interest of brevity, the explanation here omits the nonessential details. Interested readers should consult the DSE Users Manual for a more detailed explanation.

The circuit to be modeled is the common-source amplifier shown in Fig. 31(a).

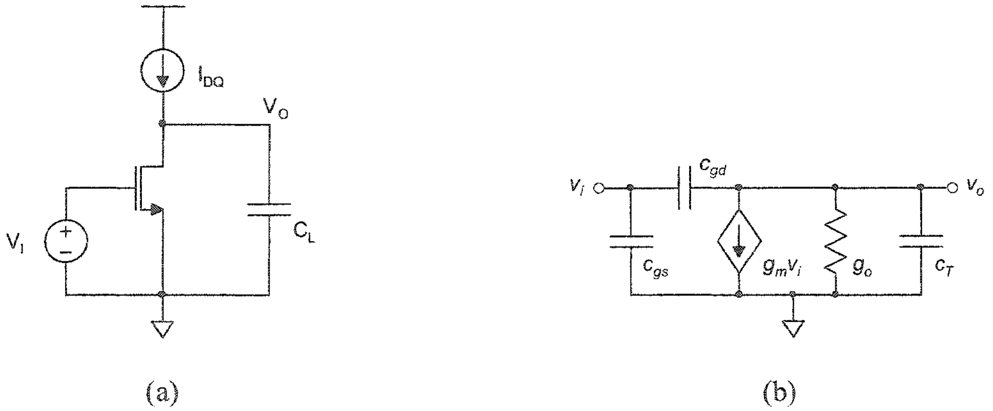


Fig. 31 Common-source amplifier (a) schematic diagram, (b) small-signal model

Choose DOFs

The first step is to identify an independent set of variables that will be referred to as *degrees of freedom* (DOF). These are the variables that the designer has control of during the design process. The design process is the method used to determine the values of these variables. Once a design's DOFs are specified, the physical circuit is uniquely defined.

There are a variety of ways to choose the DOFs. Best results are obtained by choosing a minimal set of variables. A common practice is to use physical parameters such as widths and lengths of the transistors. This, however, may not be the best approach. Since humans have an easier time conceptualizing linear relationships than nonlinear ones, it is helpful if the performance parameters are linearly related to the DOFs when possible. Since the relationship between many of the performance parameters and the excess bias (V_{EB}) is linear [43], V_{EB} was chosen as a DOF.

The DOFs chosen for this example are listed in Table 3.

Table 3 Degrees of freedom used in example

DOF	Description
V_{EB}	Excess bias voltage, $V_{EB} = (V_{GS} - V_T)$
I_{DQ}	Quiescent drain current (A)
V_{OQ}	Quiescent output voltage (V)
L	Transistor Length (m)
c_l	Capacitive load (F)

Determine expressions for the performance paramterss

The second step in the task of developing a DSE design specification file involves writing expressions for the performance parameters in terms of the DOFs. Assuming the device is in saturation, the square-law device model given in (32) applies.

$$I_{DS} = \frac{\mu \cdot C_{ox} \cdot W}{2 \cdot L} \cdot (V_{GS} - V_T)^2 \cdot (1 + \lambda \cdot V_{DS}) \tag{32}$$

Substituting the operating-point related DOFs into (32) and solving for W, yields an expression for the width of the transistor in terms of the DOFs.

$$W = \frac{2 \cdot L \cdot I_{DQ}}{\mu \cdot C_{ox} \cdot V_{EB}^2 \cdot (1 + \lambda \cdot V_{OQ})} \tag{33}$$

Note that the value of W depends upon the values of several other process-specific variables that have not yet been defined. Table 4 lists the process-related variables associated with this example. The process-related variables are not considered DOFs because once the fabrication process is chosen, they are fixed. The designer does not have the flexibility to change the process parameters during the design process.

Table 4 Process-specific variables

Name	Description
μ	Electron mobility $(\frac{m^2}{V \cdot s})$
C_{ox}	Oxide capacitance density (F/m^2)
λ	Channel length modulation parameter (V^{-1})
V_T	Threshold voltage (V) (considered constant – neglect body effect)
C_J , C_{JSW}	*N*-diffusion bottom and sidewall capacitance densities in (*F/m*^{2}) and (*F/m*) respectively.
ϕ_B	Built-in junction potential (V)
M_J , M_{JSW}	*N*-diffusion bottom and sidewall grading coefficients
ovActCont	Rule 6.2b Minimum active overlap of contact
szCont	Rule 6.1 Exact size of contact
spcContGate	Rule 6.4 Minimum spacing of contact to transistor gate
actMinWid	Rule 2.1 Minimum width of active region
polyMinWid	Rule 3.1 Minimum poly width

It is good practice to group the process-specific definitions into one file and *include* it as part of a larger design specification file. Following this method it is easy to switch from one fabrication process to another by simply *including* a different process definition file.

The small-signal model of the common-source amplifier is shown in Fig. 31(b). C_T represents the total capacitance between the output node and ground. It is composed of the fixed load capacitance *(c_L)* and the parasitic drain-bulk capacitance *(c_{db})*.

$$c_t = c_l + c_{db} \tag{34}$$

The drain-bulk capacitance is computed by (35).

$$c_{db} = \frac{C_J A_D}{\left(1 - \frac{V_F}{\phi_B}\right)^{MJ}} + \frac{C_{JSW} P_D}{\left(1 - \frac{V_F}{\phi_B}\right)^{MJSW}} \tag{35}$$

The variables A_D and P_D represent the area and perimeter of the junction, V_F is the forward bias applied to the junction (they should always remain reverse biased), and the other variables are defined in Table 4. Computing (35) is problematic because at the point in the design cycle where a designer is exploring a design space, the exact physical layout has not yet been determined. Therefore, the exact values of A_D and P_D are not available. As a result, we are forced to rely on estimates. For this example, a rectangular transistor layout like the one shown in Fig. 32 was assumed. Therefore, the drain area and perimeter can be computed using the expressions in (36) and (37) respectively.

$$A_D = W \cdot (ovActCont + szCont + spcContGate) \tag{36}$$

$$P_D = 2 \cdot (W + ovActCont + szCont + spcContGate) \tag{37}$$

Note that *ovActCont*, *szCont*, and *spcContGate* are variables that are determined by the process design rules. The meanings of these variables are described textually in Table 4 and graphically in Fig. 32.

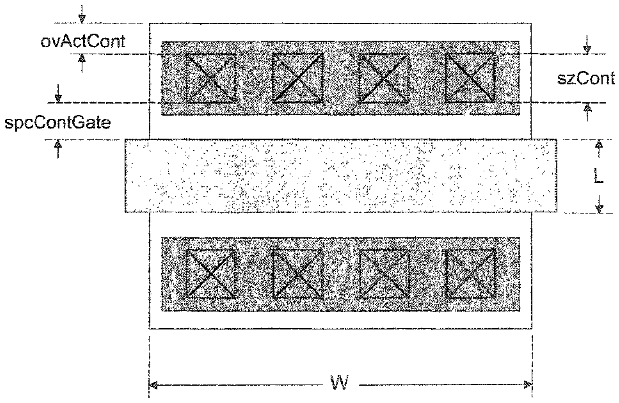


Fig. 32. Annotated rectangular transistor layout

Neglecting c_{gd}, the small-signal transfer function can be shown to be:

$$\frac{v_o}{v_i} = \frac{-g_m/g_o}{1 + s \cdot \left(\frac{c_t}{g_o}\right)} \tag{38}$$

The small-signal transconductance and output conductance are computed via (39) and (40) respectively.

$$g_m = \mu \cdot C_{ox} \cdot \frac{W}{L} \cdot V_{EB} \tag{39}$$

$$g_o = \frac{\lambda \cdot I_{DQ}}{1 + \lambda \cdot V_{OQ}} \tag{40}$$

For this example, the performance parameters of interest are the DC gain (A), the pole frequency (ω_p), and gain-bandwidth (GBW) product. Thus, the performance parameters can be computed by the expressions given in (41) thru (43).

$$A = -g_m / g_o \tag{41}$$

$$\omega_p = -g_o / c_t \tag{42}$$

$$GBW = g_m / c_t \tag{43}$$

Determine Expressions for the Constraints

The third step in the task of developing a DSE design specification file involves determining the constraints on the solution. In order for equation (32) to apply, the device must be strongly inverted and operating in the saturation region. To strongly invert the channel, the excess bias should be at least 5 times greater than the thermal voltage (roughly).

$$V_{EB} > 5 \cdot \left(\frac{k \cdot T}{q} \right) \tag{44}$$

To operate saturation, the following inequality must be satisfied.

$$V_O \geq V_{EB} \tag{45}$$

Other constraints that need to be checked are those ensure that the DOFs have reasonable values and that the transistor's width is at least as large as the minimum device width allowed in the fabrication process.

Compose Design Specification File

The final step in developing the DSE design specification file involves expressing the equations and constraints in the proper syntax. The file format is very much like XML. Each equation and constraint will be declared in separate *elements* that are delimited by *tags*. For example, the code to enter equation (33) into the design specification file is contained in Table 5.

Table 5 Code used to define the dependent parameter "*W*"

```
<EQUATION DESC="Transistor width (m)">
      W = 2 * L * IDQ / (MU * COX * VEB**2 * (1 + LAM * VOQ))
</EQUATION>
```

The equation element is delimited by an opening <EQUATION> tag and closing </EQUATION> tag. This element declares a dependent variable named *W* and assigns it a value based upon the computed value of the associated formula. A brief textual description of the variable is included as an optional argument in the opening tag. Each variable in Tables 3 and 4 and the equations (33) thru (43) are implemented in the design specification file using equation elements similar to the one shown in Table 5.

The code used to enter the constraint of (44) is contained in Table 6.

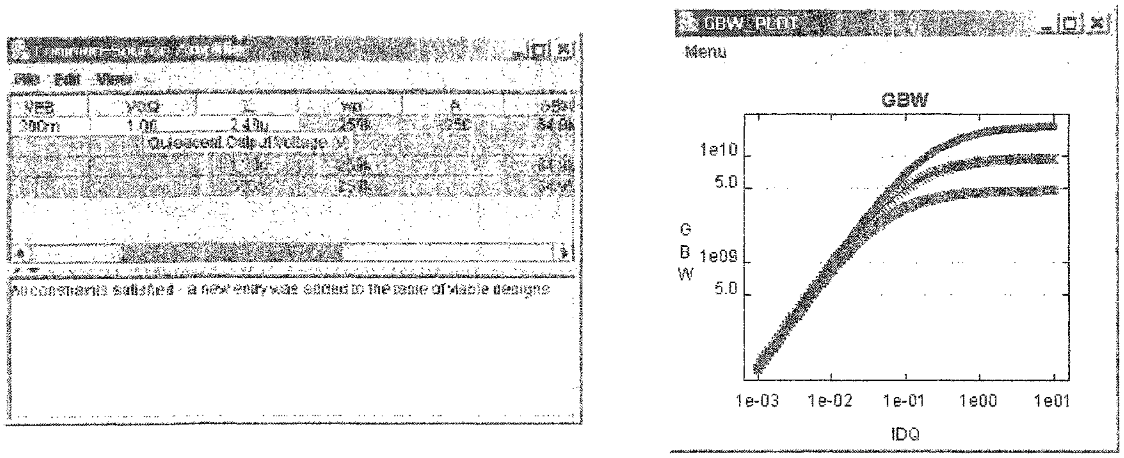
Table 6. Code used to define the constraint on the minimum V_{EB}

```
<CONSTRAINT DESC="Transistor should be strongly inverted">
      VEB > 5 * k * T / q
</CONSTRAINT>
```

The constraint element is delimited by an opening <CONSTRAINT> tag and closing </CONSTRAINT> tag. The constraint ensures the device is strongly inverted. Potential designs that do not satisfy this constraint are flagged and not entered into the table of viable designs. The other design constraints are entered into the design specification file using constraint elements that are similar to the one shown in Table 6.

Figure 33(a) shows a screen snapshot of the DSE application after loading and using the design specification file described in this example.

Design tradeoffs can be more effectively visualized by utilizing the plotting capabilities of Design Space Explorer. For example, Figure 33(b) shows the GBW as a function of IDQ for several different transistor lengths. The *plot element* that created this plot is shown in Table 7. At first glance, the code for the plot element is a little intimidating. The plot element has a lot of parameters that can be customized. Fortunately, DSE has a built-in plot-creation wizard that simplifies the task of creating plots.



All constraints satisfied - a new entry was added to the table of viable designs

(a) (b)

Fig. 33 Screen snapshots (a) after loading the example design specification file and, (b) plot showing GBW vs. I_{DQ} for transistors of various lengths

Table 7 Code used to instantiate the plot of Fig. 33

```
<Plot Name="GBW_PLOT" Title="GBW" Desc="Gain-Bandwidth Product in Hz"
XLog=true XAutoRange=true XLabel="IDQ" YLog=true YAutoRange=true
YLabel="GBW" Grid=true Legend=true Markers=true  SweepVar="IDQ"
SweepMin=0.0010 SweepMax=10.0 PointCount=60 SweepLog=true
NewEntriesVis=true >

      "GBW" = "GBW / 6.283185"

</Plot>
```

Advantages of the Network-Centric Paradigm

Network-centric applications are becoming increasingly viable as networking technology matures. In this application, several advantages accrue due to the use of the network-centric paradigm:

- Facilitates organization-wide (or worldwide) accumulation, sharing, and archiving of design knowledge by making well-documented design specification files available in a *Design Knowledge Repository*. Due to the fact that users can search the repository for circuit topologies that have already been characterized, the use of one centralized knowledge repository prevents duplication of work already performed.

- Software is available for use anytime, from any networked location, worldwide.
- Other than a Java capable web browser, no special application software is required.
- Cross-platform compatibility is assured by the use of Java. This is especially beneficial for large enterprises with multiple platforms because it helps facilitate communication of design knowledge between disparate groups.
- Since the computation occurs on the client-end, the system is scalable to support virtually any number of clients.
- Lower administration costs. Once the required Java capable browser is installed, users do not need to spend time acquiring, installing and updating software on their machine(s). Since the most recent software version resides in one place on the server, revision maintenance becomes trivial.

Conclusion

Many attempts at analog design automation focus on completing the entire design process with little or no human interaction. The lack of interactivity slows the assimilation of designer knowledge and the advancement of new circuit topologies in general. Additionally, circuit designers within an organization and throughout the world are duplicating each others work over and over again. The use of organization-wide or worldwide platform-independent design knowledge repositories offers potential for reducing the frequency of reinvention.

To address these problems, a prototype network-centric circuit design tool and design knowledge repository has been developed. It allows a designer with network access to interactively explore a circuit design space of pre-characterized circuit topologies using a convenient graphical user interface. Any user can extend the system to include new or custom circuit topologies without a lot of programming effort by writing their own design specification files.

Designers will benefit from the use of the application as it will enable them to obtain a deeper understanding of the operation of their circuits and the performance tradeoffs that are possible for pre-characterized circuit topologies. The enhanced understanding gained by the designers will result in better design realizations and will accelerate the development of improved topologies in the future.

Several benefits accrue due to the use of the network-centric computing paradigm of which the most significant is the enhanced communication of design knowledge and prevention of reinvention by the use of a centralized, platform-independent *Design Knowledge Repository*. Other benefits include reduced system administration costs, system scalability, and improved mobile computing ability.

Acknowledgements:

Support for this project has been provided in part by Raytheon Inc. and Texas Instruments Inc.

Bibliography

[35] M. G. R. Degrauwe, et al, "IDAC: An Interactive Design Tool for Analog CMOS Circuits," in *IEEE J. Solid-State Circuits*, Vol. sc-22, No. 6, pp. 1106-1116, Dec. 1987.

[36] H. Y. Koh, C. H. Sequin, and P. R. Gray, "OPASYN: A Compiler for CMOS Operational Amplifiers," in *IEEE Trans. Computer-Aided Design*, Vol. 9, No. 2, pp. 113-125, 1990

[37] R. Harjani, R. A. Rutenbar, and L. R. Carley, "OASYS: A Framework for Analog Circuit Synthesis," in *IEEE Trans. Computer-Aided Design*, Vol. 8, No. 12, pp. 1247-1266, 1989

[38] K. Swing, G. Gielen, and W. Sansen, "An Intelligent Analog IC Design System Based on Manipulation of Design Equations," in *Proc. of 1990 IEEE CICC*, pp. 8.6.1-8.6.4

[39] M. D. M. Hershenson, S. P. Boyd, and T. H. Lee, "GPCAD: a Tool for CMOS Op-Amp Synthesis," *in IEEE/ACM 1998 ICCAD Digest of Technical Papers*, pp. 296-303

[40] Analog Design Automation Inc., Ottawa, Ontario Canada, http://www.analogsynthesis.com

[41] Barcelona Design Inc., Newark, CA 94560, http://www.barcelonadesign.com

[42] Neolinear, Inc., Pittsburgh, PA 15238, http://www.neolinear.com

[43] Y. Chen, M. E. Schlarmann, and R. L. Geiger, "An Improved Design Formulation for Design and Optimization of Operational Amplifiers," *Proc. MWSCAS 1999*, Las Cruces, New Mexico, Aug 8-11, 1999, pp. 72-75

CHAPTER 5. DESIGN SPACE EXPLORER USERS MANUAL

Included as part of DSE Release 1.00

Mark E. Schlarmann and Randall L. Geiger

Introduction:

Constrained design problems are often encountered in science and engineering. These problems involve selecting values for the design degrees of freedom to achieve levels of performance that meet or exceed predefined minimum levels subject to all the applicable constraints on the solution.

Multidimensional design problems are often optimized using automated optimization routines. This involves defining a cost function and mathematically determining the values of the degrees of freedom that minimize it. The weakness in this approach is twofold. First, writing a cost-function that accurately represents a designer's priorities is a complicated task that depends upon the application under consideration. As a result, the designers are required to expend a lot of effort determining what the optimal cost function should be for each particular application. Attempting to short-circuit or approximate the cost function will result in a design that is sub-optimal. Secondly, the use of mathematical optimization tools does not substantially contribute to the designer's understanding of the relationships among the performance parameters. This is problematic because creating improved designs and design topologies is a creative process that draws upon all of a designer's experience and understanding. In fact, the use of automated optimization procedures may actually impede the development of improved designs because their non-interactive nature results in slower assimilation of designer knowledge.

Designers can rapidly deepen their understanding of a design by manually exploring a design space and observing the corresponding effects on the performance parameters. The benefits are especially clear for designs that are too complicated to derive tractable analytical relationships for the performance parameters. The additional insight into design performance gained by manual exploration of the design space will aid the designer in creating improved design topologies in the future.

This document describes a software tool that allows non-programmers to quickly create a sophisticated application for interactively exploring a design space through the use of a graphical user interface. The interactive nature of this tool will help designers more quickly gain a qualitative understanding of how the design degrees of freedom relate to the performance parameters and the design tradeoffs that are possible for a given design topology. Rather than being tailored for any specific design problem, the software is general enough to be used for the entire class of constrained design problems. Furthermore, the software is easy to use and allows users to modify existing design specifications or to create their own custom design specifications. Since it is written in *Java*, the package is compatible with the emerging network-centric computing paradigm. As such, it allows authors to publish their custom design specifications to the World-Wide-Web for users throughout the world to utilize.

Getting Started

The Design Space Explorer application can be executed as a stand-alone application or within a Java-enabled World-Wide-Web (WWW) browser as an applet.

Before we describe the operation of the software in detail, there are few items, such as how to obtain the software and how to run it, that need to be discussed. Since the software is interactive, the user needs a brief introduction to the syntax required for entering numbers and formulas.

Obtaining the Application

To use the software to interactively explore the design space of someone else's published design over the WWW, your browser should automatically download and execute the application. No user intervention should be required.

If you want to use Design Space Explorer as a stand-alone application or you want to publish your own Design Specification Files for others to use over the WWW, you need to obtain a copy of the software.

The software is distributed in a Java Archive (JAR) file called *dse.jar*. Currently the software is available along with supporting documentation from a link on the WWW page at: http://www.public.iastate.edu/~schlarm

Executing the Application

Once you have a copy of the software, it can be used as a stand-alone application for personal use and/or development of new Design Specification Files. It can also be distributed via a WWW server to publish finished design specifications on the web for global interactive use.

The application was compiled using Sun Microsystems Java Development Kit (JDK) 1.4. Therefore, it needs to execute in a virtual machine that is 1.4 compliant or better. At the present time, the virtual machines in the commercially available browsers are not yet 1.4 compliant. It is not clear that they will become compliant in the near future. In the meantime, to execute the software, the Java Runtime Environment (JRE) 1.4, which is freely available from Sun Microsystems, is required.

Publishing Design Specification Files via the WWW

Users who want to publish their own finished design specifications for others to use over the World-Wide-Web using the Design Space Explorer Applet served from their own local web-server need to obtain a copy of the Jar Archive that contains the application *dse.jar*. Place the .jar file in the same directory as the html page that references it. To ensure the browser uses the correct plug-in, the applet should be included in the html page using the code shown in Table 8.

Table 8 Code to add to the html file to include Design Space Explorer as an applet.

```
<OBJECT classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
WIDTH = 300 HEIGHT = 140
codebase="http://java.sun.com/products/plugin/autodl/jinstall-1_4-
win.cab#Version=1,4,0,0">
<PARAM NAME = CODE VALUE="edu.iastate.ee.schlarm.main.DSEApplet.class">
<PARAM NAME = ARCHIVE VALUE = "dse.jar" >
<PARAM NAME="type" VALUE="application/x-java-applet;version=1.4">
<PARAM NAME="scriptable" VALUE="false">
<PARAM NAME = "INDEXURL" VALUE ="dse_index.txt">
<COMMENT>
<EMBED type="application/x-java-applet;version=1.4" CODE =
"edu.iastate.ee.schlarm.main.DSEApplet.class" ARCHIVE = "dse.jar" WIDTH =
300 HEIGHT = 140 INDEXURL = "dse_index.txt" scriptable=false
pluginspage="http://java.sun.com/products/plugin/index.html#download">
<NOEMBED></NOEMBED>
</EMBED>
</COMMENT>
</OBJECT>
```

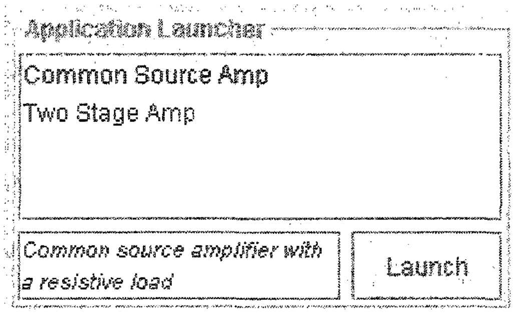
When the applet is initialized it attempts to load an index file that specifies what designs are available and where they are located. The URL of the index file is specified by the INDEXURL parameter present in the applet tag. The user can modify this parameter to point to their custom index file (notice the INDEXURL parameter tag appears twice in the code in Table 8 so the user should take care to modify it in both places). If the INDEXURL parameter tag is not present in the applet tag, then by default, the applet looks for the file named *dse_index.txt*.

An example design index file is shown in Table 9. Each line of the input file corresponds to a different Design Specification File that should be made available to the user to open using the applet. Each entry in the table consists of 3 semicolon delimited parts: the name of the Design Specification File, its URL, and a brief text description of the input file.

Table 9 Example DSF Index file

```
Common Source Amp;csa/csa.dse ;Common source amp with a resistive load
Two Stage Amp; twostageamp/twostageamp.dse; Two Stage Amplifier
```

Fig. 34 contains an example screen snapshot of what the initial applet display area might look like. The name of each design file specified in the DSF index file will be displayed in a list in the order they were defined. When the user selects one of the list items, the corresponding textual description, which was also specified in the index file, is displayed in the lower-left pane of the display. The *Launch* button starts a new Design Space Explorer Application with the currently selected Design Specification File.



Common source amplifier with a resistive load

Fig. 34 Snapshot of applet window

When Design Space Explorer is executed as an applet security restrictions are imposed on the activities that the applet can perform by the browser it is embedded in. For example, in most browsers, applets are not able to read and write directly from the file-system of the machine the applet is executing on, nor are they allowed to make network connections with any machine except the one which the applet originated from. It is also possible that on some systems applets may not be given the privilege to print. If the restrictions are too limiting in your environment, consider circumventing the security restrictions, by downloading the application and Design Specification File and running the software as a stand-alone application.

Running as a Stand-alone Application

To run Design Space Explorer as a stand-alone application obtain a copy of the Jar Archive that contains the application, *dse.jar*. Since the application is coded using the Sun Microsystems JDK 1.4 you will need a Java Runtime-Environment (JRE) that is 1.4 compliant. To launch the application use the following command in the directory where the dse.jar archive file resides:

```
…/java -jar dse.jar [ input file name]
```

where the ellipsis should be replaced by the full path to the *java* application. The *input file name* is an optional argument used to specify the name of the design specification file. If omitted, the application reads the input file from the standard input.

Number Format Specification

To interactively use the software or to develop new Design Specification Files it will be necessary to enter numeric values via a text interface. Therefore, users need to be familiar with the numeric representation the program will accept.

Real numbers are represented as integers or in exponential notation using 'e' or 'E' to designate the exponent. Examples include:

```
132

0.0

1.42e-9

-4.998E15
```

The imaginary component of a complex number is designated with a suffix of 'i' or 'I'. Examples include:

```
3.0 + 4.0i

1e8 - 5e-6I
```

Specifying Mathematical Formulas

Interactive users may find it necessary and Design Specification File developers will definitely need to enter mathematical formulas via a text interface. In order to construct valid equations, these users need to be aware of the available operations and the precedence of the operators.

All mathematical operations are computed using double (64-bit) complex math. The intrinsic operations and the order of their precedence are summarized in Table 10. Table 11 contains a list of the available built-in functions and their syntax.

Table 10 Operator precedence

Precedence	Operator	Description
1	( )	
2	**	exponentiation
3	*, /	multiplication, division
4	+, -	addition, subtraction
5	&, |, ^	bitwise and, bitwise or, bitwise xor
6	==, >, <, >=, <=, !=	equal to, greater than, less than, greater than or equal to, less than or equal to, not equal to
7	and, or, xor	logical and, logical or, logical xor

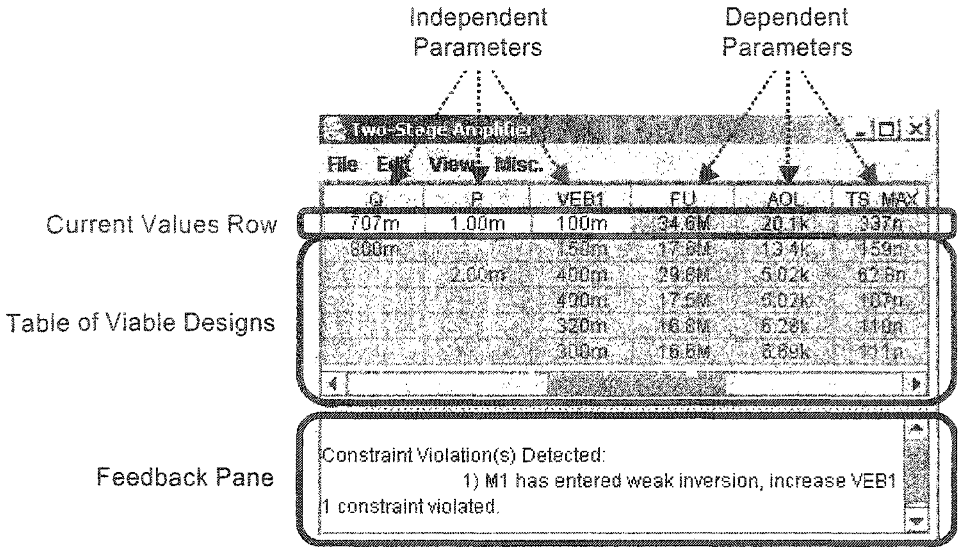
Table 11 Built-in mathematical functions

Syntax	**Function**	**Description**
abs(x)	absolute value of	returns the magnitude of x
arg(x)	argument	returns the principal angle of x measured ccw from the real axis
asin(x)	inverse sine	returns the principal inverse sine of x
asinh(x)	inverse hypebolic sine	returns the principal inverse hyperbolic sine of x
acos(x)	inverse cosine	returns the principal inverse cosine of x
acosh(x)	inverse hyperbolic cosine	returns the principal inverse hyperbolic cosine of x
atan(x)	inverse tangent	returns the principal inverse tangent of x
atanh(x)	inverse hyperbolic tangent	returns the principal inverse hyperbolic tangent of x
conj(x)	conjugate	returns the complex conjugate of x
cos(x)	cosine	returns the cosine of x (radians)
delta(x)	dirac-delta function	returns *1* if abs(x)==0, *0* otherwise
elementat(array,i)	accessing an element of an array	returns the i'th element of an array
exp(x)	exponential	returns 'e' raised to the x power
im(x)	imaginary component	returns the imaginary component of x
log(x)	natural logarithm	returns the natural logarithm of x
max(x,y)	maximum	returns the argument with the largest real component; if equal real components returns x
min(x,y)	minimum	returns the argument with the smallest real component; if equal real components returns x
polyeval(x,a0,…an)	polynomial evaluation	returns a0+a1*x^1+a2*x^2+…an*x^n

Using Design Space Explorer

Design Space Explorer Overview

Fig. 35 contains an example screen snapshot of a typical Design Space Explorer application window. The window is divided horizontally into two resizable panes. The lower pane, called the *feedback pane*, contains a text area used for providing interactive feedback to the user while the upper pane contains a table of design parameter data.



Q	P	VEB1	FU	AOL	TS_MAX
707m	1.00m	100m	34.6M	20.1k	337n
800m		150m	17.6M	13.4k	159n
	2.00m	400m	29.8M	5.02k	62.8n
		420m	17.5M	5.02k	107n
		320m	16.8M	6.28k	110n
		300m	16.6M	6.69k	111n

```
Constraint Violation(s) Detected:
                1) M1 has entered weak inversion, increase VEB1
1 constraint violated.
```

Fig. 35 Example snapshot of the Design Space Explorer Application Window

The first row in the table is a special row for editing the values of the design parameters. It is referred to as the *current values row*. The numerical values of the parameters in the first row are referred to as the *current values* of the parameters. The cells with white backgrounds are modifiable while those with gray backgrounds are either declared as constants or computed and not modifiable.

Fig. 36 shows a simplified flowchart of the operation of the tool. Most of the time the application waits for new input data. When the user enters new data, the tool recomputes all of the dependent parameters and then checks all the constraints on the solution. If any of the constraints are violated, an error message is generated and fed back to the user via the feedback pane. However, if none of the constraints were violated, the design is viable and an entry corresponding to the new design is added to the table of viable designs. Upon successful completion of the addition an informational message is generated and displayed in the feedback pane.

In Fig. 35, five viable designs were present in the table at the time the snapshot was taken. Each of the viable designs satisfies all of the applicable constraints but since their independent parameters differ, they have different values of the performance parameters.

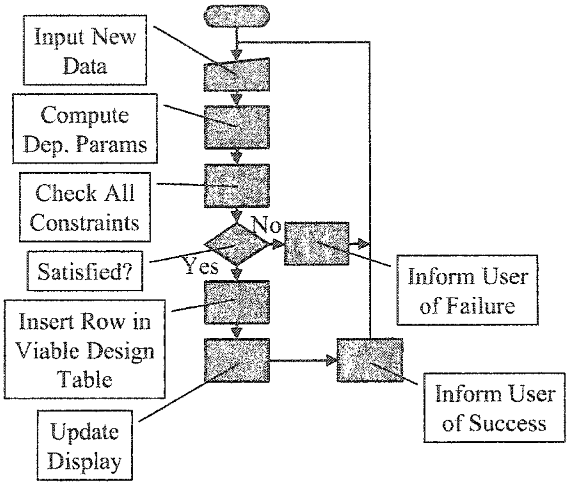


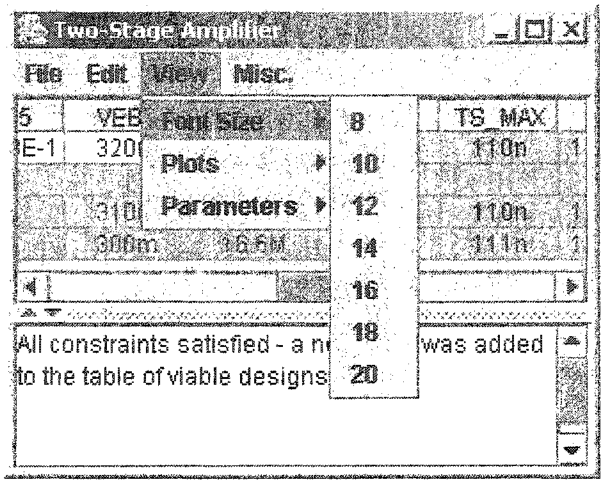
Fig. 36 Simplified operational flowchart

Special rendering techniques are employed to help users quickly identify the differences between the current design and other designs in the table. First, cells in the table of viable designs whose values are exactly the same as the current value are denoted with the "|" character. This reduces the amount of information the user has to process which allows him/her to quickly focus on the values that are different. Second, the text color is adjusted based upon the quantity to be displayed. If the real part is larger than the corresponding real component of the current value, when rounded to the desired number of significant digits for display, the value is rendered in green. Alternatively, if it is smaller, it is rendered in red. This feature help the user quickly categorize designs into those that are larger or smaller.

A particular design can be selected by mouse-clicking on its corresponding row in the table of viable designs. When a design is selected, it becomes the current design. That is, its numeric values are transferred to the current-values row. Any design can be removed from the table of viable designs by selecting it and pressing the '*delete*' key.

Changing the Font Size

By default, new windows are rendered with a 12-point font. However, the user can modify the font size at any time. As depicted in Fig. 37, clicking on **Font Size** in the **View** menu brings up a submenu of available font sizes. Clicking on one changes the font size to the selected point size throughout the application.



All constraints satisfied - a n was added
to the table of viable designs

Fig. 37 Screen snapshot while changing the font size

Controlling Parameter Visibility

When new windows are launched, the visibility and display order of the parameters in the tabular display assume the defaults specified in the Design Specification File. After launch, however, the user has complete control of parameter visibility and display order. They can change the visibility of any parameter at any time by clicking on the corresponding tree item in the selection tree contained in the Parameter Visibility Frame. The frame is activated by clicking **View->Parameter Visibility**.

For example consider the screen snapshot shown in Fig. 38 where a portion of a parameter visibility tree is visible. Each parameter in the repository is guaranteed to have at least one tree item in the tree. The checkbox preceding the parameter label indicates the current visibility status of the parameter. If checked, the parameter is currently visible. Otherwise, it is not visible. Clicking on the tree item toggles the visibility of the corresponding parameter. Note that when a parameter is made visible, it is added to the rightmost end of the table. Therefore, to see the newly added parameters, you may have to scroll the table pane all the way to the right.

Right-clicking on a category item causes a small popup menu to be displayed (like the one shown in Fig. 38). It is used to set the visibility of all the parameters in the category with a single click.

The user can modify the display order at any time by clicking on the column header of a visible table column and dragging it to the desired location.

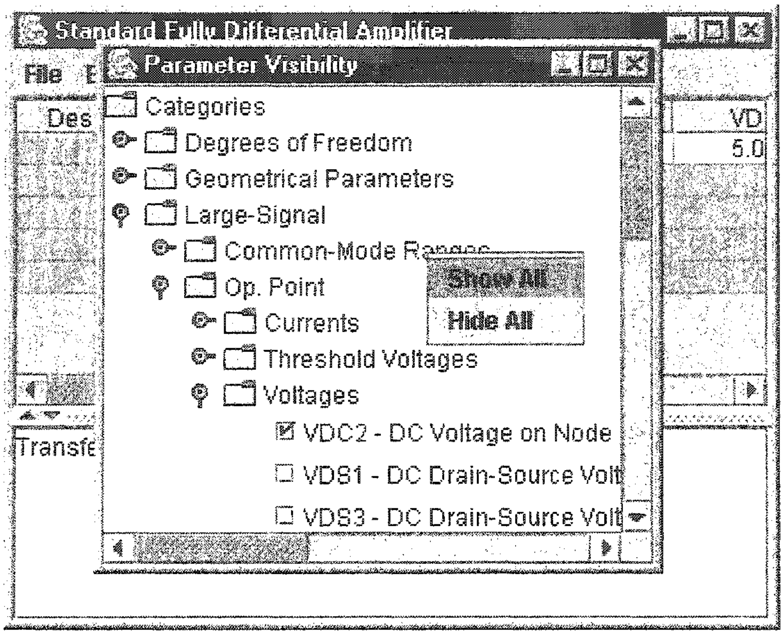


Fig. 38 Controlling parameter visibility

Printing

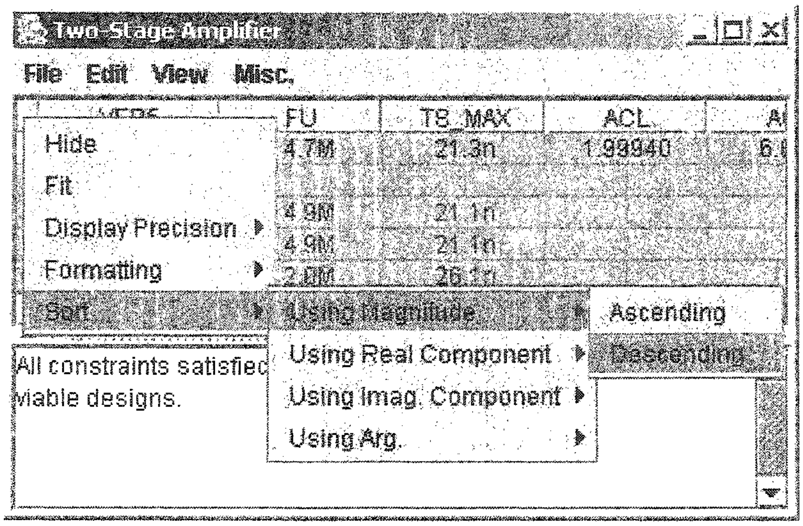
Choose **File->Print** to print the current table. If the table is too big to fit on a single sheet it will automatically be printed to multiple sheets. Tip: since the current display font size determines the print font size, changing the font size prior to printing will affect the number of pages required to print.

On some systems, due to security restrictions, when running the program as an applet, printing may not be allowed. Other systems may prompt the user for permission to print. If you are on a system which does not allow applets to print, consider downloading the code and running it as a stand-alone application. Alternatively, you might consider making the data you want to print visible and *screen-dumping* the window to the printer.

Sorting Viable Designs

As depicted in Fig. 39, clicking on a column-head in the table pops up a menu of choices associated with the selected column. One of the choices is to **Sort** the table of viable designs based on the values in one column of the table. Since the data in the table may be complex, there are a variety of options. The rows can be sorted in ascending or descending order based on the real component, the imaginary component, the argument, or the magnitude.

Since the first row is used for editing the current values of the parameters, it does not participate in the sort. The table rows are reordered based on the selected criterion using the values in the selected column as the sorting key.



FU	TS_MAX	ACL	A
4.7M	21.3n	1.99940	6.
4.9M	21.1n		
4.9M	21.1n		
2.0M	26.1n		

All constraints satisfied
viable designs.

Fig. 39 Screen snapshot depicting the column-header popup menu

Display Precision and Numeric Formatting

Clicking on a column-head in the table pops up a menu of choices associated with the selected column. Fig. 39 shows a snapshot of the popup menu. Hovering the mouse pointer over or clicking on the **Display Precision** item causes the **Digits** submenu to appear. This menu can be used to select the desired number of significant digits to be displayed. Note that this setting *only affects the precision of the display*. Despite the number of digits selected, all calculations are performed using the full precision of Java double-precision variables.

The formatting of the displayed numeric quantities can be modified in a manner that is very similar to the display precision. Hovering the mouse pointer over or clicking on the **Formatting** menu item in the column-header popup menu causes a submenu to appear that allows the numeric formatting to be set to *scientific* or *engineering* representation. In engineering notation, a character suffix is added to represent the magnitude of the quantity. Table 12 contains a list of the suffix characters and their associated magnitudes.

Table 12 Engineering notation suffixes and their magnitudes

Suffix	**Name**	**Value**
T	tera	1e12
G	giga	1e9
M	mega	1e6
k	kilo	1e3
m	milli	1e-3
u	micro	1e-6
n	nano	1e-9
p	pico	1e-12
f	femto	1e-15
a	atto	1e-18
z	zepto	1e-21
y	yotto	1e-24

Using Multiple Windows

Another feature that gives the user a great deal of flexibility in constructing a custom interface that suits their needs is the ability to simultaneously open more than one window based on the same design. You can think of this as having multiple views of the same data. Each view would have unique properties such as which parameters are visible, the order in the display, the font size, and the viable design sorting order.

To open another window click **Create New Frame** from the **File** Menu. There is no limit to the number of open frames that you can open at one time.

Row selection is synchronized among all of the open windows. For example, if you select a row in one window, even though the other open windows may have different sorting orders, the corresponding row in the other windows will also be highlighted. This feature allows easy identification of design candidates across several open windows.

Creating New Design Specification Files

To facilitate user customization of existing design specifications and encourage the development of new design specifications, the *Design Space Explorer Application* accepts plain-text design specification input files. The input files can be created with any text editor (in Windows use *Notepad*, on a Macintosh use *SimpleText*, in UNIX use *EMACS* or *VI*, or use any other word-processing program that allows you to save your document as *text*).

The input file language grammar is very similar to the extensible markup language (XML) specification. The input file is simply a list of *elements* that are denoted by *tags*. Differing types of elements represent the variety of different components that may be present in an input file. For example, one type of element represents a design equation, another type represents a constraint. There are other elements used to represent bibliographical information such as the authors' names, the title, the creation date, etc. The tags simply *mark* the beginning and ending of each element.

Typical Development Procedure

Constrained design problems are frequently encountered in science and engineering. The procedure for using it with the *Design Space Explorer* Application is summarized in the following steps.

1) Identify the design degrees of freedom (*independent parameters).*

2) Express the performance parameters (*dependent parameters*) in terms of the independent parameters and other dependent parameters with the provision that no dependency rings are created.

3) Express all solution constraints in terms of the independent and dependent parameters. Examples of typical constraints include *feasibility constraints* such as range-restrictions on signals, or *performance constraints* like minimum or maximum values for specified performance parameters.

4) Assemble the results from steps (1), (2) and (3) into an input file of the specified format described later in this document.

Tags

Tags are used to mark the beginning and ending point of an element. A simple tag consists of an opening angle bracket, (<), followed by a tag name, and then a closing angle bracket, (>). Consider the following examples <AUTHOR>, <DATE>, and <EQUATION>

Tags come in two different varieties: *starting tags* and *ending tags* and tags always occur in pairs. As suggested by their name, a starting tag marks the beginning of an element and a matching ending tag marks the end of the element. Matching starting and ending tags have identical tag names except the ending tag's tag name is prefixed by a "/". For example, the tag <AUTHOR> marks the beginning of an element while the tag </AUTHOR> would mark the end of the element.

Note: tags are not case sensitive; <DATE> is equivalent to <date>.

Starting tags can contain additional information within them which is used to customize the attributes of the element the tag is denoting. Consider the following example,

```
<CONSTRAINT DESC="Minimum Width"                                     width
> 3
</CONSTRAINT>
```

In this example, the constraint element is assigned the descriptive name "Minimum Width" by specifying it within the starting tag

Title Elements

Title elements define the title of the design specification file. This title appears in the 'about the current design spec.' dialog and the title bars of the windows.

Example:

```
      <DSFTITLE> The Sample Title </DSFTITLE>
```

Author Elements

Author elements are used to specify the names of the authors of the design specification file. There is no limit to the number of authors that can be specified. The order the authors will appear in the graphical interface will be in the same order as they appear in the design specification file.

Examples:

```
<AUTHOR> John Smith </AUTHOR>

<AUTHOR> William Jefferson Clinton </AUTHOR>
```

Revision Elements

A *revision element* is used to specify the revision number of the design specification file.

Example:

```
<REV> 1.0 </REV>
```

Date Elements

A *date element* specifies the date the design specification file was released.

Example:

```
<DATE> July 4, 1776 </DATE>
```

URL Elements

A URL element specifies the address of a WWW page that is launched when the design specification file is loaded by an applet.

Example:

```
<URL> http://www.someurl.com </URL>
```

Equation Elements

Each equation element in the input file defines a parameter. If the defined parameter depends on the value of other parameters, then it is a *dependent* parameter. Otherwise it is an *independent* parameter. Equations are declared using the following syntax:

```
<EQUATION attribute_list> var_name = math_formula </EQUATION>
```

where *attribute_list* is a white-space separated list of a combination of the relevant attributes contained in Table 13, *var_name* is the name of the parameter whose value is explicitly defined as a function of other variables and numerical constants by the equation expressed by *math_formula*.

Table 13 Attributes of the equation element

Attribute	**Description**
CATEGORIES="cat1;cat2;"	Specifies the categories the parameter defined in this equation belongs to. Note that a parameter can simultaneously be a member of more than 1 category at a time. The categories are defined in a semicolon separated list.
CONSTANT	Specifies that the user cannot interactively modify the value of the parameter.
DESC="text description"	A short textual description of the equation. Used to customize ToolTips and error messages.
HIDDEN	Sets this parameter's default visibility to be hidden. If this attribute and the visible attribute simultaneously appear in the same equation, an error will be thrown.
DIGITS=x	Specifies that *x* digits after the decimal place are to be displayed (default value = 3).
VISIBLE=ppi	Sets this parameter's default visibility to be shown with a position priority index of *ppi*. If this attribute and the hidden attribute simultaneously appear in the same equation, an error will be thrown.
NOTATION= ENG | SCI	Sets the type of notation for formatted displays. "ENG" corresponds to engineering notation while "SCI" corresponds to scientific notation.

To make the tabular graphical user interface more user friendly, the parameters can be grouped into broad categories using the CATEGORIES attribute. This allows the user to quickly perform identical actions on parameters that belong to the same category at one time rather than having to act on each one of them independently. Each parameter can simultaneously belong to more than one category. The categories are specified using a semicolon separated list.

The position priority index associated with the VISIBLE attribute is used to determine the default location of parameters in the tabular graphical user interface. Parameters with lower position priority indices are placed further to the left in the display. Parameters with equal indices are arranged alphabetically going left to right in the display.

The use of the optional, descriptive labels is highly recommended because they can greatly enhance the effectiveness of the graphical user interface by allowing for the customization of error messages, and the addition of descriptive ToolTips that are automatically displayed when the mouse rolls over specific areas of the display.

In order to function correctly, the dependency graph defined by the relationships among the equations must be *acyclic*. If a set of equations is defined that contains a dependency loop, e.g. v1 depends on v2 that depends on v1, an endless computation cycle would result. Therefore the program checks for dependency cycles and throws an exception if one is detected.

Examples:

```
<EQUATION> x = -3 + 4i </EQUATION>
```

Defines a parameter named ‘x’ and assigns it a complex value of -3+4i. Since no attributes were specified in the starting tag, ‘x’ assumes the default set of attributes. Since the right hand side of the assignment evaluates to a numerical value, with no dependence on other parameters, ‘x’ is an *independent* variable.

```
<EQUATION DESC="Underdamped Response" HIDDEN>
      y = 1 - exp( x * t )**2
</EQUATION>
```

This example defines a dependent parameter named ‘y’ that depends upon two other variables: ‘x’ and ‘t’. To customize the ToolTips and error messages associated with ‘y’ it has been assigned a short textual description of "Underdamped Response". The parameter has also been declared *hidden* meaning that, by default, it will not initially be visible in the tabular display.

```
<EQUATION
      DESC="Sampling Instant"
      CATEGORIES="Time Domain;User Modifiable"
      VISIBLE=10 DIGITS=4 NOTATION=ENG>
      t_s = 1e-5
</EQUATION>
```

This example defines a dependent variable 't_s' with a description of "Sampling Instant". The parameter is a member of the "Time Domain" and "User Modifiable" parameter categories. The VISIBLE attribute specifies that the parameter will initially be visible in the tabular display. The parameter has been assigned a position priority index (ppi) of 10 which means that all other visible parameters with ppi's less than 10 will lie to the left of 't_s' and those with ppi's greater than 10 will lie to the right in the display. Other parameters that also have ppi's of 10 will be ordered alphabetically from left to right in the display. For formatted displays, the numeric value of this parameter will be shown with four significant digits in engineering notation.

Constraint Elements

Constraint elements are declared as:

```
<CONSTRAINT attribute_list>
      comparison (( and | or | xor ) comparison )*
</CONSTRAINT>
```

where the parentheses are used for grouping only and don't really appear in the constraint definition and *attribute_list* is a white-space separated list of a combination of the relevant attributes contained in Table 14. The "|" indicates the logical 'or' operation and the '*' represents that the preceding quantity in parentheses might be repeated zero or more times. In words, a constraint element consists of a list of *comparison* items separated by the logical operators "and", "or", or "xor". The *comparison* item is defined as:

```
comparison ::= formula ( == | > | < | <= | >= | != ) formula
```

where the non-terminal *formula* is a mathematical equation that evaluates to a numerical value (see section 0). Note that the relational operators that include greater-than or less-than do not make sense for complex operands. Therefore, they are re-defined to consider only the real components of the operands only. The complex components are ignored during comparisons.

Table 14 Attributes of the constraint element

Attribute	**Description**
DESC="text description"	A short textual description of the constraint. Used to customize ToolTips and error messages.

Examples:

```
<CONSTRAINT DESC="Causality"> t >= 0 </CONSTRAINT>
```

This example defines a constraint named "Causality" that evaluates to *true* when the variable *t* is greater than or equal to zero.

```
<CONSTRAINT DESC="Stability"> re(x) < 0 </CONSTRAINT>
```

This example shows that the argument of the comparison operation can be more complicated than a single variable, it can be any expression that evaluates to a number. In this case, the constraint evaluates to *true* when the real part of *x* is less than zero.

```
<CONSTRAINT DESC="Valid Output Range">
      y < 3 and y > -3
</CONSTRAINT>
```

Defines a more sophisticated constraint that ensures a variable is within a given range (note: if y is complex, the imaginary part is ignored).

```
<CONSTRAINT DESC=“Sophisticated Constraint”>
      abs(x) == 0 or { y < 3 and y > -3 }
</CONSTRAINT>
```

Set-brackets are used to group operations at the logical operation level. Sophisticated logical functions can be compactly written by nesting logical operations. The logical expression inside the set-brackets is fully evaluated prior to performing the ‘or’ operation.

Plot Elements

Design Space Explorer has the ability to plot relationships. Plot elements are declared as:

```
<PLOT attribute_list>
      ("series_name" = "formula")+
</PLOT>
```

where *attribute_list* is a white-space separated list of a combination of the relevant attributes contained in Table 15. The ‘+’ represents that the preceding quantity in parentheses must be repeated at least once and may be repeated more than once for more than one series on the same axes. The quotation marks used to delimit the series_name are required only if the name contains whitespace characters.

Table 15 Attributes of the plot element

Attribute	**Description**
NAME="plot_name"	A unique name used to identify the plot. The name cannot contain whitespace characters. This attribute is required.
TITLE="plot title"	The plot's title
XLOG(=TRUE | FALSE)?	Flag used to indicate the independent axis should have a logarithmic scale.
XAUTORANGE(=TRUE | FALSE)?	Flag used to indicate if the independent axis should automatically rescale itself to fit the data.
XLABEL="axis_label"	Defines the independent-axis label
XMIN=X0	Used to define the minimum of the independent-axis range when XAUTORANGE is false.
XMAX=X1	Used to define the maximum of the independent-axis range when XAUTORANGE is false.
YLOG(=TRUE | FALSE)?	Flag used to indicate the dependent-axis should have a logarithmic scale.
YAUTORANGE(=TRUE | FALSE)?	Flag used to indicate if the dependent-axis should automatically rescale itself to fit the data.
YLABEL="axis_label"	Defines the dependent-axis label
YMIN=Y0	Used to define the minimum of the dependent-axis range when YAUTORANGE is false.

Table 15. Attributes of the plot element (cont.)

Attribute	**Description**
YMAX=Y1	Used to define the maximum of the dependent-axis range when YAUTORANGE is false.
GRID(=TRUE | FALSE)?	Flag used to specify the grid should be visible (default is visible).
LEGEND(=TRUE | FALSE)?	Flag used to specify if the legend should be visible (default is not visible).
MARKERS(=TRUE | FALSE)?	Flag used to specify if point-markers should be visible (default is false).
SWEEPVAR="sweep_variable_name"	Specifies the name of the independent variable that will be swept.
SWEEPMIN=S0	Used to define the minimum of the range over which the independent variable is swept.
SWEEPMAX=S1	Used to define the maximum of the range over which the independent variable is swept.
SWEEPLOG(=TRUE | FALSE)?	Flag used to indicate if the samples should be space logarithmically or linearly (default=linear spacing).
POINTCOUNT=n	Number of sample points
NEWENTRIESVIS(=TRUE | FALSE)?	Flag used to specify that when new designs are added to table of viable designs that their resultant data series be automatically visible in the plot (default = true).
DESC="text description"	A short textual description of the plot. Used to customize ToolTips and error messages.

Examples:

```
<PLOT NAME="TimeSweep" SWEEPVAR="t">

      "y(t)" = "sin(w*t + theta)"

</PLOT>
```

This example defines a plot named "TimeSweep" with a sweep variable named "t". The plot contains one data series named "y(t)" whose value is specified by the formula "sin(w*t + theta)".

```
<PLOT NAME="TimeSweep" SWEEPVAR="t">

      "ys" = "sin(w*t + theta)"

      "yc" = "cos(w*t + theta)"

</PLOT>
```

This example is the same as the previous example except that it includes two data series to be plotted on the same axes.

```
<PLOT NAME="Phase" TITLE="Phase Response" XLOG XAUTORANGE
XLABEL="Frequency" YLOG=false YAUTORANGE=false YMIN=-90 YMAX=180
YLABEL="Phase" GRID=true LEGEND=true MARKERS=false  SWEEPVAR="w"
SWEEPMIN=0.01 SWEEPMAX=1.0E9 POINTCOUNT=100 SWEEPLOG=true
NEWENTRIESVIS=true >

      "Open-Loop" = "arg(Hol)*RAD_TO_DEG"

</PLOT>
```

This is a more complex example that specifies many of the available options specified in Table 15.

Include Elements

While loading a given design specification file into the Design Space Explorer Application, another design specification file can be dynamically included (inserted) into the design specification via the *Include Statement*.

Examples:

When Design Space Explorer encounters the following statement while loading a DSF file, it inserts the contents of the file located at *c:/constants.dse* into the input stream at the point where the include statement occurred. This is an example of how an absolute path is used to incorporate an include file on a Windows-based machine.

```
<INCLUDE> c:/constants.dse </INCLUDE>
```

When Design Space Explorer is running as an applet, including files by specifying the absolute path of a file may not work due to security restrictions. In these situations, use a URL to refer to the desired include file.

```
<INCLUDE> http://www.someplace.com/constants.dse </INCLUDE>
```

Relative addressing can also be used to refer to include files. The following example looks for an include file called *constants.dse* in the same place as the design specification file that contains the include statement.

```
<INCLUDE> inc/constants.dse </INCLUDE>
```

If the previous example was encountered while loading a design specification file from a URL such as http://www.someplace.com/design.dse, Design Space Explorer would search for a file named http://www.someplace.com/inc/constants.dse. Likewise, if the previous example was encountered while loading a design specification file such as *c:/design.dse*, the application would search for *c:/inc/constants.dse.*

Filter Elements

Filter elements facilitate variable substitution in text files. They allow users to create customized reports based on the design's current values.

Menu items are added to the menu hierarchy for each filter element in the design specification file. The user activates the variable substitution process by selecting the corresponding menu item. When activated, the specified text file is processed. Each specially delimited variable placeholder is replaced with its numerical equivalent obtained from the current design.

Filter elements are declared in the design specification file as:

```
<FILTER attribute_list>
      file
</FILTER>
```

where *attribute_list* is a white-space separated list of a combination of the relevant attributes contained in Table 16 and *file* specifies the absolute or relative address of the text file to be processed.

Table 16 Attributes of the *Filter* element

Attribute	**Description**
Menu = "colon_delimited_list"	A colon delimited list describing the desired menu to place the new MenuItem on. For example "File:Create" places the new MenuItem on a submenu of the *File* menu called *Create*.
MenuText="menu text"	The text that will appear in the menu of the MenuItem used to activate the action. This is a required attribute. A parse exception will be thrown if it is not defined.
Notation="ENG|SCI"	Specifies whether all numeric quantities should be formatted in scientific or engineering notation. If not specified, each quantity is formatted according to its properties specified in the DSF file.
Digits=n	Specifies the number of significant digits used in formatting all numeric quantities. If not specified, each quantity is formatted according to its properties specified in the DSF file. *n* must be a positive integer less than or equal to 10.

Variables are delimited in the input file using *${varName}* where *varName* is the name of a properly defined variable in the design specification file. A few special variables used for date and time stamping are automatically defined. Their names are __DATE__, __TIME__, and __DATETIME__.

Example:

Table 17 lists the contents of a design specification file that declares three variables and a filter element. When this file is processed by Design Space Explorer, a menu called *create* is added to the menu bar of the application window. The *create menu* contains the menu item *Report*. When the *Report* menu item is activated, variable substitution is performed on the text file *testFilter.txt*. The contents of *testFilter.txt* are listed in Table 18. The results are listed in Table 19.

Table 17 Contents of design specification file *testFilter.dse*

```
<EQUATION> Width = 4 </EQUATION>
<EQUATION> Length = 5 </EQUATION>
<EQUATION DIGITS=6> Area = Width * Length </EQUATION>
<FILTER MENU="Create" MENUTEXT="Report"> testFilter.txt </FILTER>
```

Table 18 Contents of the text file *testFilter.txt* used to demonstrate variable substitution

```
Report Created: ${__DATETIME__}
The total area for a plot of ${Width}m X ${Length}m is ${Area} m^2.
```

Table 19 Resultant text file created by variable substitution

```
Report Created: Aug 19, 2002 11:30:10 AM
The total area for a plot of 4.00m X 5.00m is 20.0000 m^2.
```

Simple Input File Example

Up to this point in this document, each type of input file element has been independently described. Now it is time to put the pieces together into one comprehensive example that you can study. Although Design Space Explorer can be used on any constrained design problem from any discipline, the example provided here will focus on a circuit design problem. Consider the circuit schematic of the common source amplifier shown in Fig. 40.

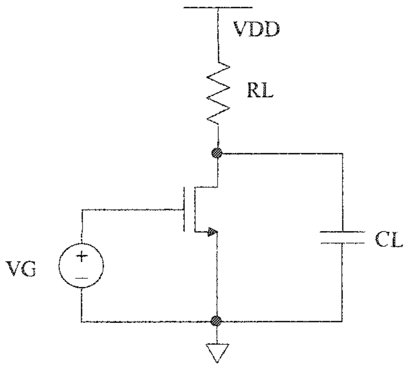


Fig. 40 Common source amplifier with a resistive load schematic

When creating a new design specification file, the first thing that is typically done is to declare the relevant bibliographical information. This involves declaring a title, the authors' names, a revision number, release date, and a URL where documentation can be found. For this example, the header information might look something like the example code shown in Table 20.

Table 20 Declaration of bibliographical information

```
<DSFTITLE>
      Single Stage Common Source Amplifier with a Resistive Load
</DSFTITLE>
<AUTHOR>
      Mark E. Schlarmann
</AUTHOR>
<REV>
      2.0
</REV>
<DATE>
      Aug. 19, 2002
</DATE>
<URL>
      http://www.public.iastate.edu/~schlarm/dsf/csa/csa.htm
</URL>
```

Let's assume the transistor of Fig. 40 is operating in the saturation region. This means we will need to impose a constraint on our solution to ensure that this is indeed the case. For a NMOS transistor to operate in the saturation region the transistor has to be *on* (its excess-bias (VEB) > 0) and its drain to source voltage (VOQ) has to exceed its excess bias. Therefore we need to add the following constraint element to the input file:

Table 21 Constraint that ensures the transistor operates in the saturation region

```
<CONSTRAINT DESC="Transistor Saturation Requirement">
      VEB > 0 and VOQ > VEB
</CONSTRAINT>
```

If we further assume the transistor operates in strong inversion, the square-law model of the transistor can be applied. Therefore, to ensure that the transistor operates in strong inversion we add the following constraint.

Table 22 Constraint that ensures the transistor operates in strong inversion

```
<CONSTRAINT DESC="Strong Inversion Requirement">
      VEB > 0.15
</CONSTRAINT>
```

When the constraints in Table 21 and Table 22 are satisfied, a square-law model similar to equation 46 can be used to model the transistor.

$$I_Q = B\frac{k_n}{2}\left(V_G - V_T\right)^2\left(1 + \lambda V_{OQ}\right) \tag{46}$$

where I_Q is the quiescent current flowing from drain to source, k_n is the product of electron mobility and the oxide capacitance density, B is the width/length ratio of the transistor, V_G is the gate voltage, V_T is the threshold voltage, λ is the channel length modulation parameter, and V_{OQ} is the quiescent output voltage.

Equation 46 can be simplified slightly by introducing an intermediate variable called the excess bias voltage (V_{EB}) which is defined as:

$$V_{EB} = V_G - V_T \tag{47}$$

Substituting (47) into (46) and solving for V_{EB} yields:

$$V_{EB} = \sqrt{\frac{2I_Q}{k_n B(1+\lambda V_{OQ})}} \tag{48}$$

Choosing V_{EB} to be a dependent (computed) parameter, (48) can be implemented by the code displayed in Table 23. Notice that V_{EB} was declared to be a member of two categories: *Dependent* and *Large Signal.*

Table 23 Code used to define the dependent variable V_{EB}

```
<EQUATION
      DESC="Excess Bias (V)"
      CATEGORIES="Dependent;Large Signal"
      HIDDEN>
      VEB = sqrt( 2 * IQ / (KN * B * (1 + LAMBDA * VOQ)))
</EQUATION>
```

In the example circuit, V_T, k_n, λ, and the supply voltage, V_{DD}, are all process-specific parameters. Once the specific fabrication process is chosen, the numerical values of these parameters are known. Table 24 contains the code used to define the values of these parameters for a hypothetical fabrication process. Notice that these parameters are each declared to be members of the *Independent* and *Process* groups and that they are each assigned a default visibility of *hidden*.

Table 24 Code used to define the process-specific parameters

```
<EQUATION
      DESC="Threshold Voltage (V)"
      CATEGORIES="Independent;Process"
      HIDDEN>
      VT = 1.0
</EQUATION>

<EQUATION
      DESC="Mobility * Cox (A/V**2)"
      CATEGORIES="Independent;Process"
      HIDDEN>
      KN = 100E-6
</EQUATION>

<EQUATION
      DESC="Channel Length Modulation Parameter"
      CATEGORIES="Independent;Process"
      HIDDEN>
      LAMBDA = 0.01
</EQUATION>

<EQUATION
      DESC="Supply Voltage (V)"
      CATEGORIES="Independent;Process"
      HIDDEN>
      VDD = 3.3
</EQUATION>
```

The parameters in equation (48) that have not yet been defined, *(I_Q, V_{OQ}, W_1, and L_1)*, are defined as independent parameters in the code shown in Table 25. In a given process under specified loading, these three variables are the primary degrees of freedom that a designer has the flexibility to change. Given their importance, these variables are by default visible and are assigned position priority indices that will ensure they are visible in the display.

Table 25 Other independent large-signal parameters

```
<EQUATION
      DESC="Quiescent Output Voltage (V)"
      CATEGORIES="Independent;Large Signal"
      VISIBLE=10>
      VOQ = 2.0
</EQUATION>

<EQUATION
      DESC="Quiescent Current (A)"
      CATEGORIES="Independent;Large Signal"
      VISIBLE=20>
      IQ=50e-6
</EQUATION>

<EQUATION
      DESC="Transistor Length (m)"
      CATEGORIES="Independent"
      VISIBLE=30>
      L1 = 10e-6
</EQUATION>

<EQUATION
      DESC="Transistor Width (m)"
      CATEGORIES="Independent"
      VISIBLE=40>
      W1 = 20e-6
</EQUATION>
```

Ignoring the parasitic gate-drain and drain-bulk capacitances, the small-signal transfer function of the circuit shown in Fig. 40 can be written as:

$$H(s) = \frac{-g_m}{g_d + {}^{1}\!/_{R_L} + sC_L} \tag{49}$$

The code required to calculate the response at a particular frequency is contained in Table 26.

Table 26 Evaluation of the transfer function at a particular frequency

```
<EQUATION                                                                   HIDDEN
      CATEGORIES="Independent;Small Signal"
      DESC="Evaluation frequency (Hz)">
      f = 1.0
</EQUATION>

<EQUATION                                                                   HIDDEN
      CATEGORIES="Dependent;Small Signal"
      DESC="Evaluation frequency (rad/s)">
      w = 2 * 3.14159265359 * f
</EQUATION>

<EQUATION                                                                   HIDDEN
      CATEGORIES="Dependent;Small Signal"
      DESC="Complex Frequency Variable">
      s = (0+1i)*w
</EQUATION>

<EQUATION
      DESC="Open-Loop Transfer Function"
      CATEGORIES="Dependent;Small Signal"
      HIDDEN>
      Hol = - gm / (gd + 1/RL + s*CL)
</EQUATION>
```

Using the open-loop transfer function given in (49), it is possible to derive expressions for several small signal performance parameters including the DC gain, the bandwidth, and the gain-bandwidth product. The code required to implement these expressions in the design specification file is shown in Table 27.

Table 27 Small signal performance parameters

```
<EQUATION
      DESC="DC Gain"
      CATEGORIES="Dependent;Performance;Small Signal"
      VISIBLE=40>
      A0 = - gm / (gd + 1/RL)
</EQUATION>

<EQUATION
      DESC="Bandwidth (Hz)"
      CATEGORIES="Dependent;Small Signal;Performance"
      VISIBLE=50>
      BW =  ( 1/RL + gd ) / ( 2 * CL * 3.141592653 )
</EQUATION>

<EQUATION
      DESC="UGF or Gain-Bandwidth (Hz)"
      CATEGORIES="Dependent;Small Signal;Performance"
      VISIBLE=60>
      UGF = A0 * BW
</EQUATION>
```

The small signal performance parameters in Table 27 are defined in terms of the small signal transconductance g_m, the channel conductance g_d, the load resistance R_L, and the load capacitance, C_L. The transconductance and channel conductance can each be derived from the large signal parameters already defined as shown in Table 28. Since g_m and g_d are intermediate quantities used to derive the performance parameters of interest, we have chosen to make them hidden by default.

Table 28 Code used to derive the small signal parameters from the large signal parameters

```
<EQUATION
      DESC="Transconductance"
      CATEGORIES="Dependent;Small Signal"
      HIDDEN>
      gm = KN * B * VEB
</EQUATION>

<EQUATION
      DESC="Channel Conductance"
      CATEGORIES="Dependent;Small Signal"
      HIDDEN>
      gd = LAMBDA * IQ
</EQUATION>
```

The quiescent output current, quiescent output voltage, and supply voltage were each defined as independent variables earlier in Tables 24 and 25. Therefore, we know the current flowing through R_L and the voltage across it. The required size of R_L can be computed using Ohm's Law as shown in the code present in Table 29.

Table 29 Required load resistance in terms of the independent variables

```
<EQUATION
      DESC="Load Resistance"
      CATEGORIES="Dependent"
      VISIBLE=35>
      RL = (VDD - VOQ) / IQ
</EQUATION>
```

The load capacitance C_L is an independent parameter that is really determined by the application under consideration. Once the application is identified, the loading is usually specified. The code in Table 30 is used to define the load capacitance.

Table 30 Code used to define the load capacitance C_L

```
<EQUATION
      DESC="Load Capacitance (F)"
      CATEGORIES="Independent"
      HIDDEN>
      CL = 20e-12
</EQUATION>
```

Power consumption is another performance parameter that is often of interest to designers. The code in Table 31 demonstrates how we can add a power consumption performance parameter to the design specification file. Other performance parameters could be defined in a similar manner.

Table 31 Definition of the power consumption performance parameter

```
<EQUATION
      DESC="Power Consumption (W)"
      CATEGORIES="Dependent;Small Signal;Performance"
      VISIBLE=70>
      P = IQ * VDD
</EQUATION>
```

Since it has not previously been defined and we will need it to restrict the solution space to viable solutions only, an expression for the gate voltage is declared as shown in Table 32.

Table 32 Expression used to compute the gate voltage

```
<EQUATION
      DESC="DC Gate Voltage (V)"
      CATEGORIES="Dependent;Large Signal"
      HIDDEN>
      VG = VEB + VT
</EQUATION>
```

Some additional constraints are required to eliminate solutions that are not viable due to voltage range limitations on the transistor's gate and the output node. These voltage range constraints are shown in Table 33.

Table 33 Additional constraints on solution due to limited gate voltage range

```
<CONSTRAINT DESC="Quiescent Output Voltage Range Restriction">
      VOQ < VDD and VOQ > 0.0
</CONSTRAINT>

<CONSTRAINT DESC="Acceptable Gate Voltage Range">
      VG < VDD and VG > VT
</CONSTRAINT>
```

The code shown in Table 34 implements a plot of the open-loop magnitude response similar to the one shown in Fig. 41.

Table 34 Code used to plot the magnitude response

```
<Plot
      Name="Magnitude Response"
      Title="Open-Loop Magnitude Response"
      Desc="Open-Loop Magnitude Response"
      XLog=true
      XAutoRange=true
      XLabel="Frequency (Hz)"
      YLog=true
      YAutoRange=true
      YLabel="Magnitude Response"
      Grid=true
      Legend=false
      Markers=true
      SweepVar="f"
      SweepMin=100
      SweepMax=1.0E8
      PointCount=60
      SweepLog=true
      NewEntriesVis=false >
      "Hol_mag" = "abs(Hol)"
</Plot>
```

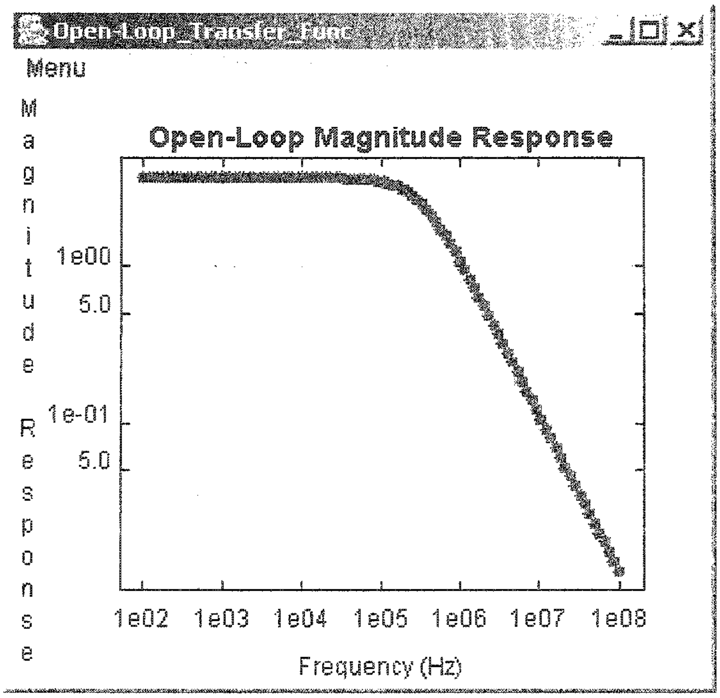


Fig. 41 Open-loop magnitude response

The filter element of Table 35 is included to allow the user to create a customized SPICE netlist based on the design's current values. This element adds a menu called "*create*" to the menubar and places a user selectable menu item named "*WinSpice Netlist*" in it. When the user activates the menu item, the application parses the file "*netlist.txt*" and substitutes numerical values for symbolic variables present in the file. The resultant SPICE netlist is presented in a new editor pane. An example netlist file is included as Table 36.

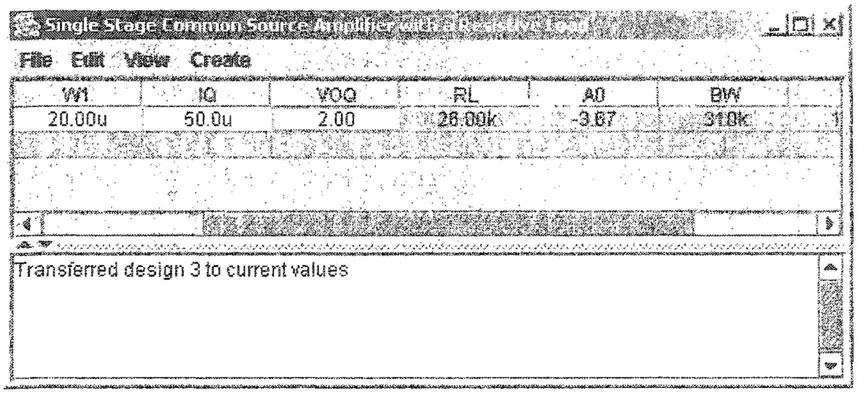
Table 35 Filter element used to allow the user to create a customized SPICE netlist

```
<FILTER
      MENU="Create"
      MENUTEXT="WinSpice Netlist">
      netlist.txt
</FILTER>
```

Table 36 SPICE netlist containing symbolic variables

```
Common-Source Amplifier with Resistive Load
*
* WinSpice Netlist
* Created: ${__DATETIME__}
*
* Estimated Performance:
*      DC Gain    = ${A0}
*      Power (W)  = ${P}
*      UGF   (Hz) = ${UGF}
*      BW    (Hz) = ${BW}
*
VDD ndd 0 DC ${VDD}
VSS nss 0 DC 0.0
VGS in nss DC ${VGS} AC 1.0
RL ndd out ${RL}
CL out nss ${CL}
M1 out in nss nss CMOSN w=${W1} l=${L1}
*
.MODEL CMOSN NMOS LEVEL=1 VTO=${VT} KP=${KN} LAMBDA=${LAMBDA}
*
.control
destroy all
save all
op
reset
ac dec 50 1000 1e8
plot vdb(out)
print vm(out)
.endc
*
.END
```

That concludes the example. Of course, this design specification can easily be modified or extended to fit your particular needs. Using this same approach you can develop design specification files for your own custom applications. For reference, the entire design specification code listing has been included in Table 37. Fig. 42 shows a screen snapshot of the resulting application window after loading the "comprehensive example" design specification file contained in Table 37. Notice the bibliographic information specified in the input file is visible in the feedback pane and the parameters are ordered in the display based upon their position priority indexes.



W1	IQ	VOQ	RL	A0	BW
20.00u	50.0u	2.00	28.00k	-3.87	310k

```
Transferred design 3 to current values
```

Fig. 42 Screen snapshot after loading the “comprehensive example”

Table 37 Entire Design Specification File for the comprehensive example

```
<<DSFTITLE>                                                                   Single
Stage Common Source Amplifier with a Resistive Load
</DSFTITLE>

<AUTHOR>
      Mark E. Schlarmann
</AUTHOR>

<REV>
      2.0
</REV>

<DATE>
      Aug. 19, 2002
</DATE>

<URL>
      http://www.public.iastate.edu/~schlarm/dsf/csa/csa.htm
</URL>

<CONSTRAINT DESC="Transistor Saturation Requirement">
      VEB > 0 and VOQ > VEB
</CONSTRAINT>

<CONSTRAINT DESC="Strong Inversion Requirement">
      VEB > 0.15
</CONSTRAINT>

<EQUATION
      DESC="Excess Bias (V)"
      CATEGORIES="Dependent;Large Signal"
      HIDDEN>
      VEB = sqrt( 2 * IQ * L1 / (KN * W1 * (1 + LAMBDA * VOQ)))
</EQUATION>

<EQUATION
      DESC="Threshold Voltage (V)"
      CATEGORIES="Independent;Process"
      HIDDEN>
      VT = 1.0
</EQUATION>

<EQUATION
      DESC="Mobility * Cox (A/V**2)"
      CATEGORIES="Independent;Process"
      HIDDEN>
      KN = 100E-6
</EQUATION>

<EQUATION
      DESC="Channel Length Modulation Parameter"
      CATEGORIES="Independent;Process"
      HIDDEN>
      LAMBDA = 0.01
</EQUATION>
```

Table 37. Entire Design Specification File for the comprehensive example (Cont.)

```
<EQUATION
      DESC="Supply Voltage (V)"
      CATEGORIES="Independent;Process"
      HIDDEN>
      VDD = 3.3
</EQUATION>

<EQUATION
      DESC="Quiescent Output Voltage (V)"
      CATEGORIES="Independent;Large Signal"
      VISIBLE=10>
      VOQ = 2.0
</EQUATION>

<EQUATION
      DESC="Quiescent Current (A)"
      CATEGORIES="Independent;Large Signal"
      VISIBLE=20>
      IQ=50e-6
</EQUATION>

<EQUATION
      DESC="Transistor Width (m)"
      CATEGORIES="Independent"
      VISIBLE=30
      DIGITS=4>
      W1 = 20e-6
</EQUATION>

<EQUATION
      DESC="Transistor Length (m)"
      CATEGORIES="Independent"
      VISIBLE=40
      DIGITS=4>
      L1 = 10e-6
</EQUATION>

<EQUATION
      HIDDEN
      CATEGORIES="Independent;Small Signal"
      DESC="Evaluation frequency (Hz)">
      f = 1.0
</EQUATION>

<EQUATION
      HIDDEN
      CATEGORIES="Dependent;Small Signal"
      DESC="Evaluation frequency (rad/s)">
      w = 2 * 3.14159265359 * f
</EQUATION>

<EQUATION
      HIDDEN
      CATEGORIES="Dependent;Small Signal"
      DESC="Complex Frequency Variable">
      s = (0+1i)*w
</EQUATION>
```

Table 37. Entire Design Specification File for the comprehensive example (Cont.)

```
<EQUATION
      DESC="Open-Loop Transfer Function"
      CATEGORIES="Dependent;Small Signal"
      HIDDEN>
      Hol = - gm / (gd + 1/RL + s*CL)
</EQUATION>

<EQUATION
      DESC="DC Gain"
      CATEGORIES="Dependent;Performance;Small Signal"
      VISIBLE=50>
      A0 = - gm / (gd + 1/RL)
</EQUATION>

<EQUATION
      DESC="Bandwidth (Hz)"
      CATEGORIES="Dependent;Small Signal;Performance"
      VISIBLE=60>
      BW =  ( 1/RL + gd ) / ( 2 * CL * 3.141592653 )
</EQUATION>

<EQUATION
      DESC="UGF or Gain-Bandwidth (Hz)"
      CATEGORIES="Dependent;Small Signal;Performance"
      VISIBLE=70>
      UGF = sqrt(gm**2 - (gd + 1/RL)**2)/( 2 * CL * 3.141592653 )
</EQUATION>

<EQUATION
      DESC="Transconductance"
      CATEGORIES="Dependent;Small Signal"
      HIDDEN>
      gm = KN * W1 * VEB * ( 1 + LAMBDA * VOQ ) / L1
</EQUATION>

<EQUATION
      DESC="Channel Conductance"
      CATEGORIES="Dependent;Small Signal"
      HIDDEN>
      gd = LAMBDA * IQ / ( 1 + LAMBDA * VOQ )
</EQUATION>

<EQUATION
      DESC="Load Resistance"
      CATEGORIES="Dependent"
      VISIBLE=45
      DIGITS=4>
      RL = (VDD - VOQ) / IQ
</EQUATION>

<EQUATION
      DESC="Load Capacitance (F)"
      CATEGORIES="Independent"
      HIDDEN>
      CL = 20e-12
</EQUATION>
```

Table 37. Entire Design Specification File for the comprehensive example (Cont.)

```
<EQUATION
      DESC="Power Consumption (W)"
      CATEGORIES="Dependent;Small Signal;Performance"
      VISIBLE=70>
      P = IQ * VDD
</EQUATION>

<EQUATION
      DESC="DC Gate-Source Voltage (V)"
      CATEGORIES="Dependent;Large Signal"
      HIDDEN>
      VG = VEB + VT
</EQUATION>

<CONSTRAINT DESC="Quiescent Output Voltage Range Restriction">
      VOQ < VDD and VOQ > 0.0
</CONSTRAINT>

<CONSTRAINT DESC="Acceptable Gate Voltage Range">
      VG < VDD and VG > VT
</CONSTRAINT>

<PLOT
      Name="Magnitude_Response"
      Title="Open-Loop Magnitude Response"
      Desc="Open-Loop Magnitude Response"
      XLog=true
      XAutoRange=true
      XLabel="Frequency (Hz)"
      YLog=true
      YAutoRange=true
      YLabel="Magnitude Response"
      Grid=true
      Legend=false
      Markers=true
      SweepVar="f"
      SweepMin=100
      SweepMax=1.0E8
      PointCount=60
      SweepLog=true
      NewEntriesVis=false>
      "Hol_mag" = "abs(Hol)"
</PLOT>

<FILTER
      MENU="Create"
      MENUTEXT="WinSpice Netlist">
      netlist.txt
</FILTER>
```

Acknowledgments

Special thanks to Sandy Anderson of Middlesex University and Priyantha Jayanetti of the University of Maine for their contribution of the Complex number class used for this project. It is publicly available at www.netlib.org.

Additional thanks go to Edward A. Lee and Christopher Hylands at U. C. Berkeley for the Ptplot package from which the plotting routines used in this project were derived. The Ptplot package is publicly available at http://ptolemy.eecs.berkeley.edu/java/ptplot/.

Appendix A. Source Code

Please see the supplemental CD-ROM for copies of the source code and documentation.

CHAPTER 6. MODELING AMPLIFIERS USING DESIGN SPACE EXPLORER

This document describes the development of Design Space Explorer (DSE) models for a few commonly-used amplifier topologies including a standard single-stage differential amplifier, the telescopic-cascode amplifier, and a two-stage amplifier. Extrapolating the basic procedures outlined here, models for other structures can easily be developed.

Prior to discussing the particular amplifier architectures, some introductory material will be presented to provide the necessary background, describe some of the assumptions, and introduce the notation used. Next, the model of each amplifier architecture will be described in detail.

Common Material

This introductory section contains material that is common to each of the amplifier architectures that will be considered. Putting this material up front circumvents the need to duplicate information in later sections of the document.

Degrees of Freedom

The usual procedure for developing a DSE Model involves identifying an independent set of variables referred to as *degrees of freedom* (DOF) and expressing the performance parameters in terms of those variables. Best results are obtained by choosing a minimal set of variables for the DOFs. Since humans have an easier time conceptualizing linear relationships than nonlinear ones, it is also helpful if the performance parameters are linearly related to the DOFs where possible.

In the general case, most designs require a large number of degrees of freedom to fully characterize the possible design space. However, in a specific application, the actual number of variables the designer may have the freedom to vary may be much less.

For example, using a top-down design methodology, several design parameters are determined by the requirements of the system the amplifier will be used in. These parameters are usually related to interfacing with the other components of the system. Examples might include the common-mode input and output ranges or the capacitive load on the amplifier. The values of these parameters may be dictated by needs of the other components in the system and are typically agreed upon prior to the actual design of the amplifier. For a given design, these parameters are typically set at the beginning of the design process and not modified again. In this case, the general model defines the common-mode ranges and the capacitive load to be degrees of freedom. However, in the specific application described, these variables are fixed and not available for change during the actual design process.

Since the variables that are fixed vary from application to application, rather than designing the DSE model for a specific application's requirements, it is good practice to try to keep the models as general as possible. It is also good practice to make all the DOFs visible by default. This allows the user to assign values to the DOFs whose values are dictated by the application upon startup. Then he/she can *hide* the static parameters which eliminates their display in the graphical user interface. Following this approach ensures the fixed parameters are assigned the proper value and helps to focus the designer's attention on the design's actual degrees of freedom.

Large-Signal Device Models

Large-signal models relate a device's terminal currents to its terminal voltages. These models are required to find a circuit's operating point.

Design Space Explorer is not meant to be a replacement for SPICE. On the contrary, it allows a designer to more quickly explore the achievable design tradeoffs prior to committing a design to a full SPICE simulation. Simple device models are employed to avoid obscuring the procedure with too much complexity. Designers interested in more accurate models can modify the examples presented here to include the second-order effects of interest.

The square-law model was employed to model strongly inverted FETs operating in saturation. Equations (50) and (51) contain the square law equations for NMOS and PMOS devices respectively.

$$I_{DSN} = \frac{\mu_n C_{oxn} W}{2L_{eff}}\left(V_{GS} - V_{TN}\right)^2\left(1 + \lambda_n V_{DS}\right) \tag{50}$$

$$I_{SDP} = \frac{\mu_p C_{oxp} W}{2L_{eff}}\left(V_{SG} - V_{TP}\right)^2\left(1 + \lambda_p V_{SD}\right) \tag{51}$$

The parameters in the formulae have their usual values. They are described in Table 38.

Table 38 Parameters of the large-signal device models

Name	**Description**
W	Device width (m)
L_{eff}	Effective device length (m) = L_{drawn} – 2 * LD
V_{GS}, V_{SG}	Gate-source and source-gate voltages respectively (V)
V_{TN}, V_{TP}	NMOS and PMOS threshold voltages respectively (V) (considered constant – ignore body effect)
λ_n, λ_p	Channel length modulation parameters $\left(V^{-1}\right)$
V_{DS}, V_{SD}	Drain-source and source-drain voltages respectively (V)
μ_n, μ_p	Electron and hole mobility respectively ($\frac{m^2}{V \cdot s}$)
C_{oxn}, C_{oxp}	NMOS and PMOS oxide capacitance densities $\left(\frac{F}{m^2}\right)$ Calculated via ε_{OX}/T_{OX}

The multiplicative constant coefficients are lumped into single coefficients given by:

$$\beta_n = \frac{\mu_n C_{oxn}}{2} \tag{52}$$

and

$$\beta_p = \frac{\mu_p C_{oxp}}{2} \tag{53}$$

As a result, (50) and (51) can be expressed more succinctly as:

$$I_{DSN} = \beta_n \frac{W}{L_{eff}} \left(V_{GS} - V_{TN}\right)^2 \left(1 + \lambda_n V_{DS}\right) \tag{54}$$

and

$$I_{SDP} = \beta_p \frac{W}{L_{eff}} (V_{SG} - V_{TP})^2 (1 + \lambda_p V_{SD}) \tag{55}$$

Contrary to usual convention, as they are used in (55), V_{TP} and λ_p are both defined to be positive quantities.

Excess Bias Voltage

For a FET, the threshold voltage is the minimum gate-source potential that causes an inversion layer to form under the gate. The FET does not conduct until the gate-source voltage exceeds the threshold voltage. Increasing the gate-source voltage beyond the threshold increases the conductivity of the device, turning the device on harder. As a convenient measure of how hard a particular device is turned on, a term called the *overdrive* or *excess bias voltage* is often used. These terms simply refer to the amount by which the gate-source voltage exceeds the device threshold. For an NMOS transistor,

$$V_{EBN} = V_{GS} - V_T \tag{56}$$

where V_{GS} is the gate-source voltage and V_T is the threshold voltage. Remembering the polarity of a signed quantity can be difficult when you have many different variables you are working with. Therefore, some practitioners prefer to work only with positive quantities. For this reason, the excess bias for a PMOS is sometimes defined as the negative of (56) and is given by:

$$V_{EBP} = -(V_{GS} - V_T) \tag{57}$$

Using these definitions, for both NFETs and PFETs, the devices are on when their excess bias voltages are positive.

Ensure Operation in Strong Inversion

No clear demarcation between weak and strong-inversion exists. Simple analytical models have been developed for transistors operating in both regions. Weak inversion models apply when excess bias voltages are very small while strong-inversion models are appropriate for large excess biases. In the transition region though, there is some question as to exactly which model should be used. Clearly, using the strong-inversion model for very small excess bias voltages would be inappropriate. For this reason, operation in strong inversion is assumed and minimum excess bias voltage constraints are placed on the solution to ensure that the strong inversion model is not used under weak inversion conditions.

$$V_{EB} > VEB\_MIN \tag{58}$$

There are differing opinions as to exactly how large VEB_MIN should be. Some feel 5~6 thermal voltages is enough. Others believe a lower value is sufficient. For this reason, VEB_MIN has been established as a user-modifiable parameter. By default, it is set to the conservative value of 150 mV and can be changed if desired. Constraints are added design specification file to ensure that (58) is satisfied for each device.

Ensure Operation in Saturation

Except in low-power or extremely low-voltage applications, the devices in the amplifiers that will be considered are normally biased to operate in saturation. For this reason, large-signal models appropriate for the saturation region were assumed. To ensure the models are not used in cases where one or more of the devices are not saturated, constraints were added to the design specification file. For each NFET and PFET, constraints of the form of (59) and (60) are imposed respectively.

$$V_{DSN} \geq V_{EB} \tag{59}$$

$$V_{SDP} \geq V_{EB} \tag{60}$$

Small-Signal Parameters

The small-signal frequency dependent models used for the FETs are shown in Fig. 43. Since an n-well process was assumed, the NMOS and PMOS models differ slightly.

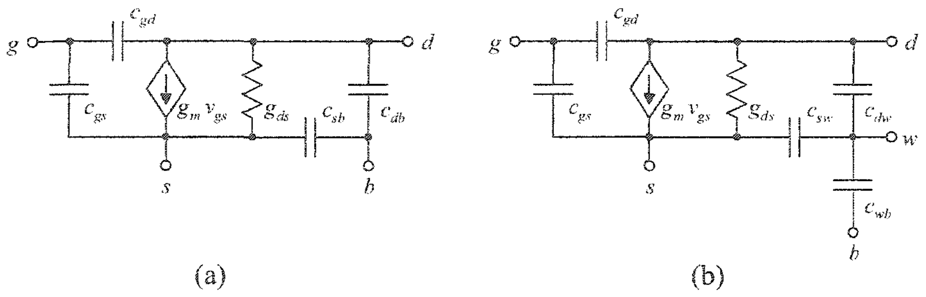


Fig. 43 Frequency dependent small-signal models of (a) NMOS and (b) PMOS transistors

The transconductance parameter is obtained by differentiating the large-signal expression for the current with respect to the gate-source voltage. For an NMOS transistor:

$$g_{mn} = \frac{\partial I_{DSN}}{\partial V_{GS}} = \frac{2\beta_n W}{L}(V_{GS} - V_{TN})(1 + \lambda_n V_{DS}) = \frac{2I_{DSN}}{V_{GS} - V_{TN}} \tag{61}$$

Similarly, for a PMOS transistor:

$$g_{mp} = \frac{\partial I_{SDP}}{\partial V_{SG}} = \frac{2\beta_p W}{L}(V_{SG} - V_{TP})(1 + \lambda_p V_{SD}) = \frac{2I_{SDP}}{V_{SG} - V_{TP}} \tag{62}$$

The output conductance is the partial derivative of the large-signal current with respect to the drain-source voltage. For an NMOS transistor:

$$g_{dn} = \frac{\partial I_{DSN}}{\partial V_{DS}} = \lambda_n \beta_n \frac{W}{L}(V_{GS} - V_{TN})^2 = \frac{\lambda_n I_{DSN}}{1 + \lambda_n V_{DS}} \tag{63}$$

For a PMOS transistor:

$$g_{dp} = \frac{\partial I_{SDP}}{\partial V_{SD}} = \lambda_p \beta_p \frac{W}{L}(V_{SG} - V_{TP})^2 = \frac{\lambda_p I_{SDP}}{1 + \lambda_p V_{SD}} \tag{64}$$

The process of estimating the sizes of the parasitic capacitors is described next.

Estimating FET Parasitics

Although the specific physical layout of the circuit is required to accurately estimate the circuit parasitics, we can make a few assumptions to obtain reasonable estimates of their sizes. Fig. 44 graphically depicts the device parasitics that are relevant for FETs operating in saturation in an n-well process.

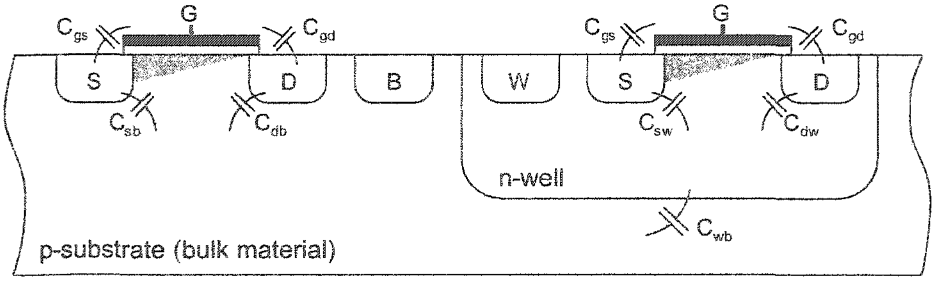


Fig. 44 Relevant parasitic capacitances for FETs operating in saturation

The gate-drain capacitance, C_{gd} is a fixed-geometry capacitor. It has a value of:

$$C_{gd} = C_{ox} W L_D \tag{65}$$

where C_{ox} is the thin oxide capacitance density, W is the device width, and L_D is the lateral diffusion of the drain under the gate.

The gate-source capacitance is given by:

$$C_{gs} = C_{ox} W L_D + \frac{2}{3} C_{ox} W L \tag{66}$$

The first term corresponds to gate-source overlap due to the lateral diffusion of the source region. The second term corresponds to the distributed capacitance that exists between the gate and channel.

The drain-bulk, drain-well, and well-bulk capacitors are junction capacitors. Their effective capacitance is operating point dependent and modeled by:

$$C_{db}, C_{dw}, C_{wb} = \frac{C_J A_D}{\left(1 - \frac{V_F}{\phi_B}\right)^{MJ}} + \frac{C_{JSW} P_D}{\left(1 - \frac{V_F}{\phi_B}\right)^{MJSW}} \tag{67}$$

where C_J and C_{JSW} are the bottom and sidewall capacitance densities, A_D and P_D are the area and perimeter of the junction, V_F is the forward bias applied to the junction (the junctions are almost always reverse biased), ϕ_B is the built-in potential associated with the junction, M_J and M_{JSW} are the bottom and sidewall junction grading coefficients. Please note that the coefficients for a PMOS and NMOS transistor differ. Therefore, although C_{db} and C_{dw} are of the same functional form, under equal forward bias, their values will differ.

Since the exact physical layouts of the devices are not available, estimates of the parasitics are required. To make reasonable estimates, knowledge of the layout style and the process design rules are required. Therefore, several constants have been defined in the DSE file. Table 39 lists the constants and their meanings. Their default values were obtained from the MOSIS CMOS scalable submicron (SCMOS-SUBM) design rules using the alternative contact to active. For use with other design rules including vendor specific rules, these definitions will need modification. For more detailed information on the design rules refer to http://www.mosis.org.

Table 39 Design rules constants available for estimating parasitics

Name	Value	Description
LAM (?)	0.3μ	Half the minimum process feature size
spcWellDiffPot	18?	Rule 1.2 Minimum spacing between wells at different potential
actMinWid	3?	Rule 2.1 Minimum width of active region
spcAct	3?	Rule 2.2 Minimum spacing of active regions of same type
spcSDActWell	6?	Rule 2.3 Minimum spacing source/drain active to well edge
spcSWContWell	3?	Rule 2.4 Substrate/well contact active to well edge
spcPActNAct	4?	Rule 2.5 Minimum spacing between non-abutting active of different implants
polyMinWid	2?	Rule 3.1 Minimum poly width
spcPolyOverAct	3?	Rule 3.2a Minimum poly spacing over active
szCont	2?	Rule 6.1 Exact size of contact
ovActCont	?	Rule 6.2b Minimum active overlap of contact
spcContGate	2?	Rule 6.4 Minimum spacing of contact to transistor gate

One might assume that if the bulk and source diffusions are the same physical size, that the bulk-source and bulk-drain capacitance would be the same. This is not the case. The bulk-source capacitor is larger because a portion of the distributed channel to bulk capacitance is lumped together and assigned to the bulk-source capacitance.

$$C_{sb}, C_{sw} = \frac{C_J A_D}{\left(1 - \frac{V_F}{\phi_B}\right)^{MJ}} + \frac{C_{JSW} P_D}{\left(1 - \frac{V_F}{\phi_B}\right)^{MJSW}} + \frac{\left(\frac{2}{3}\right) C_J W (L - 2L_D)}{\left(1 - \frac{V_F}{\phi_B}\right)^{MJ}} \tag{68}$$

$$= \frac{C_J\left(A_D + \frac{2W(L-2L_D)}{3}\right)}{\left(1-\frac{V_F}{\phi_B}\right)^{MJ}} + \frac{C_{JSW}P_D}{\left(1-\frac{V_F}{\phi_B}\right)^{MJSW}} \quad (69)$$

A conductive channel exists under the gate of a device in saturation. As a result, a reversed-biased junction forms at the interface of the channel and the bulk material. Therefore, a junction capacitance is present between the bulk and the channel. Since there is no channel node in our device models, as an approximation, two-thirds of the distributed channel to bulk capacitance is assigned to the bulk-source capacitance. The third term of (68) accounts for the distributed channel to bulk capacitance.

Computational Efficiency Issues

Design Space Explorer does not optimize the mathematical expressions used to compute the parameters. Therefore, the computational efficiency of Design Space Explorer can be improved by defining quantities that are computed more than once as intermediate variables. For example, suppose *Y1* and *Y2* are defined as shown in (70) and (71)

$$Y1 = A + B + C \quad (70)$$

$$Y1 = A + B + D \quad (71)$$

Y1 and *Y2* can be computed more efficiently by the expressions shown in (72)-(74). An intermediate variable, *T*, was introduced to store the value of *A*+*B*.

$$Y1 = T + C \quad (72)$$

$$Y2 = T + D \quad (73)$$

$$T = A + B \quad (74)$$

Since higher levels of abstraction are obtained through the use of intermediate variables, the task of writing design specification files is made easier through their use as well.

Symmetric Response

Applying arbitrary input signals $\{v_1, v_2\}$ to the linear 2-port circuit of Fig. 45 elicits a response that will be denoted as $\{y_1, y_2\}$.

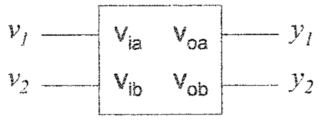


Fig. 45 Two-port circuit excited by $\{v_1, v_2\}$ results in $\{y_1, y_2\}$

Using an arrow to represent the transformation,

$$\begin{Bmatrix} v_1 \\ v_2 \end{Bmatrix} \rightarrow \begin{Bmatrix} y_1 \\ y_2 \end{Bmatrix} \tag{75}$$

The system is said to have a ***symmetric response*** if swapping the inputs results in a reversal of the output signals.

$$\text{if } \begin{Bmatrix} v_2 \\ v_1 \end{Bmatrix} \rightarrow \begin{Bmatrix} y_2 \\ y_1 \end{Bmatrix}, \text{ then } \underline{\text{response is symmetric}} \tag{76}$$

Fully-differential amplifiers are electrically symmetric. Assuming perfect device matching, their responses are also symmetric. This property will be exploited to show that the total small-signal response of a fully-differential amplifier can be decomposed into separate common-mode and differential-mode components.

The Total Small-Signal Response of a Fully-Differential Amplifier

Suppose that arbitrary small-signal input signals $\{v_{ia}, v_{ib}\}$ are applied to a fully-differential amplifier. The resultant small-signal response will be designated as $\{v_{oa}, v_{ob}\}$.

$$\begin{Bmatrix} v_{ia} \\ v_{ib} \end{Bmatrix} \rightarrow \begin{Bmatrix} v_{oa} \\ v_{ob} \end{Bmatrix} \tag{77}$$

The input signals can be equivalently expressed in terms of their common-mode and differential-mode components.

$$\begin{Bmatrix} v_{icm} + v_{id}/2 \\ v_{icm} - v_{id}/2 \end{Bmatrix} \rightarrow \begin{Bmatrix} v_{oa} \\ v_{ob} \end{Bmatrix} \tag{78}$$

Since the small-signal model is a linear approximation of the nonlinear response, superposition applies. The input is decomposed into four separate signals as shown in (79). The responses that would result if each of these signals were independently applied to the system are computed. The total response is the appropriately weighted summation of those responses.

$$\begin{Bmatrix} v_{icm} \\ 0 \end{Bmatrix} + \begin{Bmatrix} 0 \\ v_{icm} \end{Bmatrix} + \frac{1}{2}\begin{Bmatrix} v_{id} \\ 0 \end{Bmatrix} - \frac{1}{2}\begin{Bmatrix} 0 \\ v_{id} \end{Bmatrix} \rightarrow \begin{Bmatrix} v_{oa} \\ v_{ob} \end{Bmatrix} \tag{79}$$

Supposė the first term in (79) is independently applied to the system. The resultant response will be designated as $\{y_1, y_2\}$.

$$\begin{Bmatrix} v_{icm} \\ 0 \end{Bmatrix} \rightarrow \begin{Bmatrix} y_1 \\ y_2 \end{Bmatrix} \tag{80}$$

As discussed in 0, fully-differential amplifiers are symmetric and therefore have symmetric responses. As a result, the response due to the second term of (79) is given by:

$$\begin{Bmatrix} 0 \\ v_{icm} \end{Bmatrix} \rightarrow \begin{Bmatrix} y_2 \\ y_1 \end{Bmatrix} \tag{81}$$

The portion of the response due to common-mode input variations is found by summing (80) and (81).

$$\begin{Bmatrix} v_{icm} \\ v_{icm} \end{Bmatrix} \rightarrow \begin{Bmatrix} y_1 + y_2 \\ y_1 + y_2 \end{Bmatrix} \tag{82}$$

Designate the response of the system that would result if the third parenthesized term of (79) were applied as $\{y_3, y_4\}$.

$$\begin{Bmatrix} v_{id} \\ 0 \end{Bmatrix} \rightarrow \begin{Bmatrix} y_3 \\ y_4 \end{Bmatrix} \tag{83}$$

Since the response is symmetric, application of the fourth term would result in a response of:

$$\begin{Bmatrix} 0 \\ v_{id} \end{Bmatrix} \rightarrow \begin{Bmatrix} y_4 \\ y_3 \end{Bmatrix} \tag{84}$$

The portion of the response due to differential variations is found by subtracting (84) from (83).

$$\frac{1}{2}\begin{Bmatrix} v_{id} \\ -v_{id} \end{Bmatrix} \rightarrow \frac{1}{2}\begin{Bmatrix} (y_3 - y_4) \\ -(y_3 - y_4) \end{Bmatrix} \tag{85}$$

The total response is the summation of (82) and (85).

$$\begin{Bmatrix} v_{icm} + v_{id}/2 \\ v_{icm} - v_{id}/2 \end{Bmatrix} \rightarrow \begin{Bmatrix} (y_1 + y_2) + (y_3 - y_4)/2 \\ (y_1 + y_2) - (y_3 - y_4)/2 \end{Bmatrix} \tag{86}$$

The common-mode and differential components of the output response can be computed as shown in (87) and (88) respectively.

$$v_{ocm} = \frac{v_{oa} + v_{ob}}{2} = (y_1 + y_2) \tag{87}$$

$$v_{od} = v_{oa} - v_{ob} = (y_3 - y_4) \tag{88}$$

Substituting (87) into (82) yields the response due to common-mode input variations.

$$\begin{Bmatrix} v_{icm} \\ v_{icm} \end{Bmatrix} \rightarrow \begin{Bmatrix} v_{ocm} \\ v_{ocm} \end{Bmatrix} \tag{89}$$

Substituting (88) into (85) yields the response to differential input signals.

$$\frac{1}{2}\begin{Bmatrix} v_{id} \\ -v_{id} \end{Bmatrix} \rightarrow \frac{1}{2}\begin{Bmatrix} v_{od} \\ -v_{od} \end{Bmatrix} \tag{90}$$

Observe from (89) that common-mode variations at the input result in common-mode output variations but do not result in differential variations at the output. From (90) it is evident that differential input variations result in differential output variations but do not result in common-mode output variations. Thus, amplifiers with symmetric responses do not exhibit common-mode to differential-mode or differential-mode to common-mode conversion (This is, in fact, not exactly true because the small-signal model is a linear approximation to a nonlinear set of equations that is only approximately valid for small-signal variations).

Summing (89) and (90) yields the total small-signal response.

$$\begin{Bmatrix} v_{ia} \\ v_{ib} \end{Bmatrix} = \begin{Bmatrix} v_{icm} + v_{id}/2 \\ v_{icm} - v_{id}/2 \end{Bmatrix} \rightarrow \begin{Bmatrix} v_{oa} \\ v_{ob} \end{Bmatrix} = \begin{Bmatrix} v_{ocm} + v_{od}/2 \\ v_{ocm} - v_{od}/2 \end{Bmatrix} \tag{91}$$

Expressed in the Laplace Domain, the relationship between v_{ocm} and v_{icm} can be written as:

$$v_{ocm}(s) = H_{cm}(s) \cdot v_{icm}(s) \tag{92}$$

where $H_{cm}(s)$ is the common-mode transfer function. Like the common-mode component, the relationship between v_{id} and v_{od} can be expressed in the Laplace Domain as:

$$v_{od}(s) = H_d(s) \cdot v_{id}(s) \tag{93}$$

where $H_d(s)$ is the differential-mode transfer function. Since the Laplace Transform is a linear operator, the right-hand side of (91) can be expressed in the Laplace Domain as:

$$\begin{Bmatrix} L\{v_{oa}\} \\ L\{v_{ob}\} \end{Bmatrix} = \begin{Bmatrix} L\{v_{ocm}\} + L\{v_{od}\}/2 \\ L\{v_{ocm}\} - L\{v_{od}\}/2 \end{Bmatrix} \tag{94}$$

$$\begin{Bmatrix} v_{oa}(s) \\ v_{ob}(s) \end{Bmatrix} = \begin{Bmatrix} v_{ocm}(s) + v_{od}(s)/2 \\ v_{ocm}(s) - v_{od}(s)/2 \end{Bmatrix} \tag{95}$$

Substituting (92) and (93) into the right-hand side of (95) yields frequency domain expressions for the total response in terms of the common-mode and differential-mode transfer functions.

$$\begin{Bmatrix} v_{oa}(s) \\ v_{ob}(s) \end{Bmatrix} = \begin{Bmatrix} H_{cm}(s) \cdot v_{icm}(s) + H_d(s) \cdot v_{id}(s)/2 \\ H_{cm}(s) \cdot v_{icm}(s) - H_d(s) \cdot v_{id}(s)/2 \end{Bmatrix} \tag{96}$$

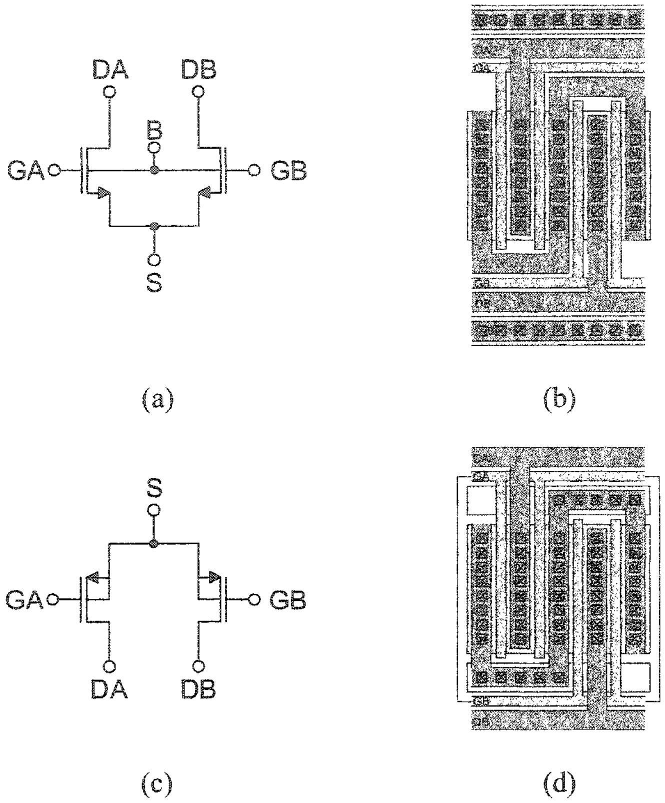
Transistor Layouts

In order to develop a Design Space Explorer Model that accurately represents the available design tradeoffs, the circuit parasitics need to known. However, at the point in the design cycle where a designer is exploring a design space, the exact physical layout has not yet been determined. For this reason, we rely on estimates of the parasitics during design space exploration.

Specific transistor layout styles are assumed for each transistor in the design. Based on these assumptions, reasonable estimates of the parasitics are obtained.

Differential Pairs

In differential amplifiers, the transistors in the differential pair need to accurately match each other. Because they obtain accurate matching in the presence of process gradients, interdigitized common-centroid layouts are commonly used to realize differential pairs. Fig. 46 shows NMOS and PMOS differential pairs and examples of basic building-blocks that can be used to produce their interdigitized common-centroid layouts.

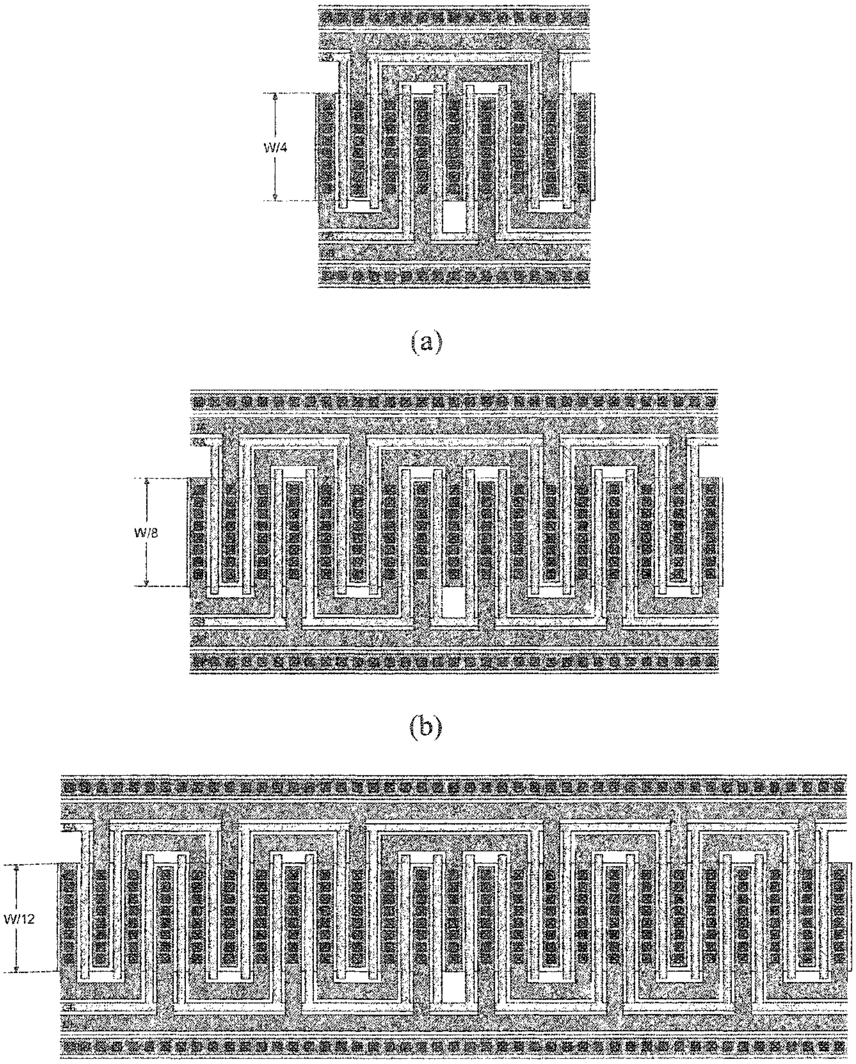


(a) (b)

(c) (d)

Fig. 46 Differential pairs (a) NMOS diff. pair (b) NMOS building-block
(c) PMOS diff. pair (d) PMOS building-block

An interdigitized common-centroid layout based on one of the building-blocks of Fig. 46 is constructed by the following procedure. One or more instances of the building-block are connected together side-by-side to form a higher-level block referred to as block #1. A second block, block #2, is created by mirroring block #1 about a vertical line (horizontally flipping). Block #1 and block #2 are connected to form the final structure. Fig. 47 shows examples of NMOS differential pair layouts created using the technique.



(a)

(b)

(c)

Fig. 47 Interdigitized common-centroid NMOS differential pair layouts (a) ABBA sequence (b) ABABBABA sequence (c) ABABABBABABA sequence

For these layouts, the sensitivity to process gradients is minimized by choosing the interdigitization sequence that results in the "*squarest*" aspect ratio. In special cases, other factors, such as packing density, may compel the designer to choose an aspect ratio that is not equal to one. The aspect ratio of the active area of an interdigitized common-centroid layout based on the building blocks of Fig. 46 can be expressed as shown in (97).

$$AR = \frac{W_{active}}{H_{active}} = \frac{2 \cdot NF \cdot (L + \beta) + \beta}{(W/NF)} \qquad (97)$$

The variable *AR* refers to the aspect ratio, *L* is the transistor length, *W* is the transistor width, *NF* is the number of fingers per transistor, and β is given by (98).

$$\beta = szCont + 2 \cdot spcContGate \tag{98}$$

The variables *szCont* and *spcContGate* refer to the size of a contact and the minimum spacing between a contact and a transistor's gate respectively. Their values are specified along with other process-specific constants in Table 39. Given knowledge of the desired aspect ratio, the appropriate number of fingers/FET can be determined by solving (97) for *NF*. The result is given in (99).

$$NF' = \alpha \cdot \left(\sqrt{1 + \frac{2 \cdot W \cdot DAR}{\alpha \cdot \beta}} - 1 \right) \tag{99}$$

The variable *DAR* represents the desired aspect ratio, and α is given in (100).

$$\alpha = \frac{\beta}{4(L + \beta)} \tag{100}$$

The quantity *NF'* varies continuously with the other parameters. However, using the building-block approach requires that the layout consists of an even number of blocks. Since each block has two fingers per transistor, the number of fingers must be quantized to an integer multiple of 4. Therefore, the actual number of fingers per FET is computed by (101).

$$NF = \max\left(4 \cdot \mathrm{rint}\left(\frac{NF'}{4} \right), 4 \right) \tag{101}$$

The function *rint()* rounds its argument to the nearest integer. The *max(x,y)* function ensures that no layout has fewer than four fingers/FET.

The areas and perimeters of the drains are given by (102) and (103) respectively.

$$A_D = \frac{W \cdot \beta}{2} \tag{102}$$

$$P_D = W + NF \cdot \beta \tag{103}$$

To avoid edge-effects, dummy transistors are usually placed on each end of the interdigitized common-centroid array. Therefore, the areas and perimeters of the sources are related to the drain areas and perimeters by (104) and (105) respectively.

$$A_S = 2 \cdot \left(\frac{NF+1}{NF} \right) A_D \tag{104}$$

$$P_S = 2 \cdot \left(\frac{NF+1}{NF} \right) P_D \tag{105}$$

Equations (104) and (105) compute the area and perimeter of the joint source diffusion. Since there are two transistors, half of the area and perimeter is usually assigned to each of the transistors. An n-well process is assumed; therefore PMOS transistors need to lie in a well. Assuming the array is padded with a dummy transistor on each end, the minimum width and height of the well that can accommodate the interdigitized array is given by (106) and (107) respectively.

$$W_{well} = 2 \cdot \left[(NF+1)(L+\beta) + ovActCont + spcSWContWell \right] + szCont \tag{106}$$

$$H_{well} = \frac{W}{NF} + 2 \cdot (spcPActNAct + szCont + 2 \cdot ovActCont + spcSWContWell) \tag{107}$$

The constants *ovActCont*, *spcPActNAct*, and *spcSWContWell* are the process-specific constants defined in Table 39 and *L* is the transistor length. The corresponding well perimeter and area is calculated by (108) and (109) respectively.

$$P_{well} = 2 \cdot (W_{well} + H_{well}) \tag{108}$$

$$A_{well} = W_{well} H_{well} \tag{109}$$

The minimum transistor width that can be laid-out without design rule violations depends upon the number of fingers in the transistor. The relationship is given by:

$$W_{\min} = NF \cdot actMinWid \tag{110}$$

The value of the constant *actMinWid* is the minimum width of poly. Its value is specified along with the other process-specific constants in Table 39. Constraints are added to the design specification file to ensure that the minimum width requirements are met for each transistor.

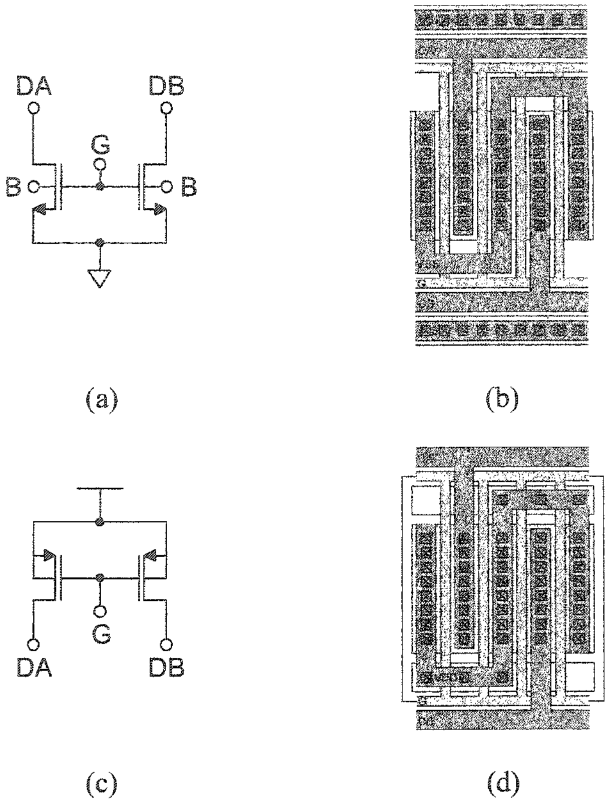


Fig. 48 Matched current sources (a) NMOS current source (b) NMOS building-block (c) PMOS current source (d) PMOS building-block

Matched Current Sources

If the tail node of a differential pair is connected to the appropriate power rail and the gate terminals are tied together, the differential pair degenerates into a set of matched current sources. Therefore, with only minor modifications, the same building blocks used for realizing interdigitized common-centroid differential pair layouts can be used to realize matched current sources. Fig. 48 shows the schematic diagrams and basic-building blocks used to realize interdigitized common-centroid layouts of matched NMOS and PMOS current sources.

The resultant structures have the same drain/source/well areas and perimeters as the differential pair interdigitized common-centroid layouts. Therefore, equations (97)-(110) are valid for these structures as well.

Single Transistors

Fingered layouts are more compact and can have smaller drain parasitics when compared to a conventional rectangular layout. Therefore, when possible, fingered layouts are preferred. Fig. 49 shows examples of basic building-blocks that can be used to produce fingered layouts of NFETs and PFETs whose bulks are tied to their sources.

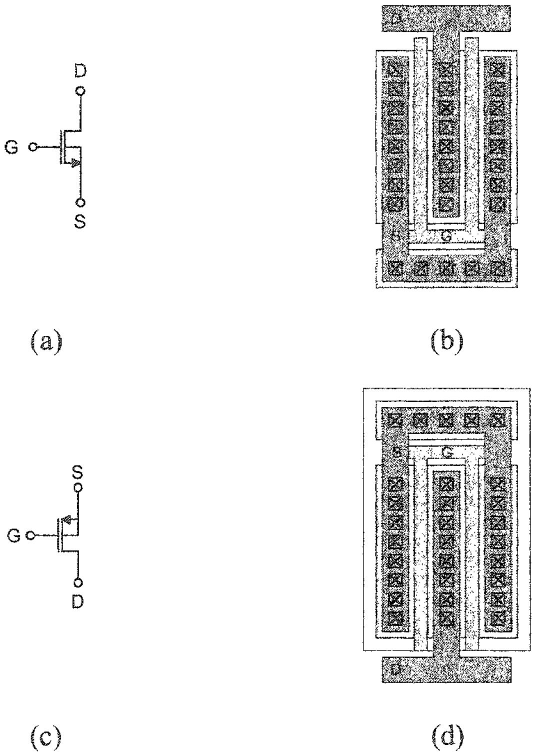
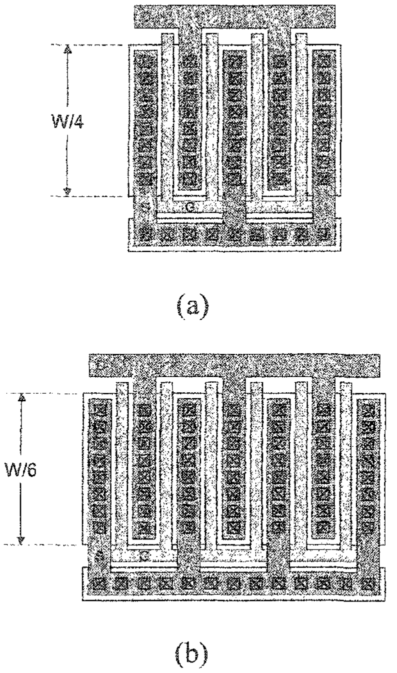


Fig. 49 Fingered layout of single transistors with bulks tied to their sources (a) NMOS schematic (b) NMOS building-block (c) PMOS schematic (d) PMOS building-block

A fingered layout based on one of the building blocks of Fig. 49 is constructed by connecting one or more of the building blocks side-by-side. No flipping or rotating is required. To illustrate, Fig. 50 shows several examples of fingered NFETs that were created using the building block of Fig. 49(a).



(a)

(b)

Fig. 50 Fingered NFET layouts (a) 4 fingers (b) 6 fingers

The aspect ratio of the active area of a fingered layout constructed using one of the building blocks of Fig. 49 can be expressed as shown in (111).

$$AR = \frac{W_{active}}{H_{active}} = \frac{NF \cdot (L + \beta) + \beta}{(W/NF)} \tag{111}$$

The quantity β is a constant given by (98), W is the transistor width; L is the transistor length, and NF is the number of fingers in the FET.

Given the desired aspect ratio, the appropriate number of fingers/FET is determined by solving (111) for NF. The result is given in (112).

$$NF' = 2 \cdot \alpha \cdot \left(\sqrt{1 + \frac{W \cdot DAR}{\alpha \cdot \beta}} - 1 \right) \tag{112}$$

The variable DAR represents the desired aspect ratio, and α is given in (100). The quantity NF' varies continuously with the other parameters. However, using the building-block approach requires that the layout consists of an integer number of blocks. Since each block has two fingers per transistor, the number of fingers must be quantized to an integer multiple of 2. Therefore, the actual number of fingers per FET is computed by (113).

$$NF = \max\left(2 \cdot \mathrm{rint}\left(\frac{NF'}{2}\right), 2\right) \tag{113}$$

The areas and perimeters of the drains are given by (114) and (115) respectively.

$$A_D = \frac{W \cdot \beta}{2} \tag{114}$$

$$P_D = W + NF \cdot \beta \tag{115}$$

To avoid edge-effects, dummy transistors are usually placed on each end of the array. Therefore, the areas and perimeters of the sources are related to the drain areas and perimeters by (116) and (117) respectively.

$$A_S = \left(\frac{NF+2}{NF}\right) A_D \tag{116}$$

$$P_S = \left(\frac{NF+2}{NF}\right) P_D \tag{117}$$

An n-well process is assumed; therefore PMOS transistors need to lie in a well. Assuming the array is padded with a dummy transistor on each end, the minimum width and height of the well that can accommodate the fingered layout is given by (118) and (119) respectively.

$$W_{well} = (NF+2)(L+\beta) + \beta + 2 \cdot spcSDActWell \tag{118}$$

$$H_{well} = \frac{W}{NF} + spcPActNAct + szCont + 2 \cdot ovActCont + 2 \cdot spcSWContWell \tag{119}$$

The corresponding well perimeter and area is calculated by (120) and (121) respectively.

$$P_{well} = 2 \cdot (W_{well} + H_{well}) \tag{120}$$

$$A_{well} = W_{well} H_{well} \tag{121}$$

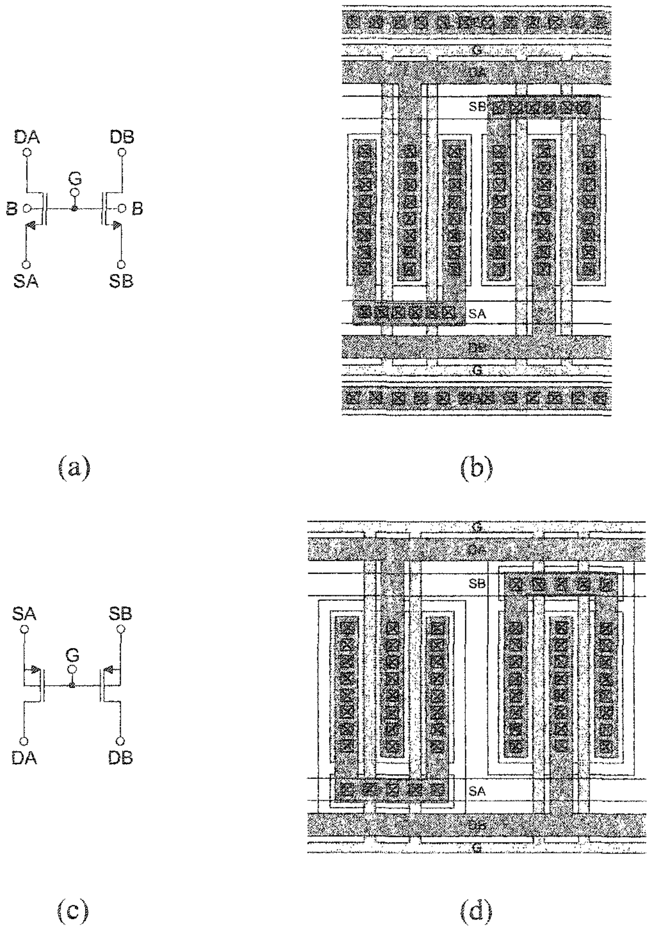
The minimum transistor width that can be laid-out without design rule violations depends upon the number of fingers in the transistor. The relationship is given by:

$$W_{\min} = NF \cdot actMinWid \tag{122}$$

The value of the constant *actMinWid* is the minimum width of an active region. Its value is specified along with the other process-specific constants in Table 39. Constraints are added to the design specification file to ensure that the minimum width requirements are met for each transistor.

Matched Common-Gate Pairs (Cascodes)

In amplifier design, cascode transistors are commonly used. For differential amplifiers, pairs of cascode transistors need to match as accurately as possible. Therefore, interdigitized common-centroid layouts are commonly employed. Fig. 51 shows NMOS and PMOS matched common-gate pairs (cascodes) and examples of basic building-blocks that can be used to produce their interdigitized common-centroid layouts.



(a) (b)

(c) (d)

Fig. 51 Matched common-gate pairs (a) NMOS schematic (b) NMOS building-block (c) PMOS schematic (d) PMOS building-block

The procedures for constructing interdigitized common-centroid layouts using the building blocks of Fig. 51 are the same as those described for the differential pair in section 0. The width of the smallest rectangle that totally encloses the active region of an interdigitized common-centroid layout of a common-gate pair is of the form:

$$W_{active} = \frac{NF}{2} \cdot (\mathrm{K} + \kappa) - \kappa \tag{123}$$

where *NF* is a variable whose value specifies the number of fingers per transistor. K and κ are constants expressed, in part, in terms of the design rule constants given in Table 39. K is given in (124) while κ's value depends on the transistor type. If the transistors are n-type, expression (125) applies, otherwise for p-types, use (126).

$$\mathrm{K} = 2 \cdot (ovActCont + \beta + L) + szCont \tag{124}$$

The quantity β is a constant given by (98) and L is the transistor length.

$$\kappa_{nmos} = spcAct \tag{125}$$

$$\kappa_{pmos} = 2 \cdot spcSDActWell + spcWellDiffPot \tag{126}$$

The height of the minimum-sized rectangle that encloses the active region of an interdigitized common-centroid layout of a common-gate pair is:

$$H_{active} = \frac{W}{NF} \tag{127}$$

Given expressions (123) and (127), the aspect ratio of the enclosing rectangle is written as:

$$AR = \frac{W_{active}}{H_{active}} = \frac{(\mathrm{K} + \kappa) \cdot NF^2 - 2 \cdot \kappa \cdot NF}{2 \cdot W} \tag{128}$$

Given a desired aspect ratio *(DAR)*, (128) can be solved for the number of fingers per transistor.

$$NF' = \xi \left\{ 1 + \sqrt{1 + \frac{2 \cdot W \cdot DAR}{\kappa \cdot \xi}} \right\} \tag{129}$$

The variable ξ is the ratio given in (130).

$$\xi = \frac{\kappa}{\mathrm{K} + \kappa} \tag{130}$$

The quantity *NF'* varies continuously with the other parameters. However, using the building-block approach requires that the layout consists of an even number of blocks. Since each block has two fingers per transistor, the number of fingers must be quantized to an integer multiple of 4. Therefore, the actual number of fingers per FET is computed by (101).

$$NF = \max\left(4 \cdot \mathrm{rint}\left(\frac{NF'}{4}\right), 4\right)$$

The function *rint()* rounds its argument to the nearest integer. The *max(x,y)* function ensures that no layout has fewer than four fingers/FET.

The areas and perimeters of the drains are given by (131) and (132) respectively.

$$A_D = \frac{W \cdot \beta}{2} \tag{131}$$

$$P_D = W + NF \cdot \beta \tag{132}$$

The areas and perimeters of the sources are given by (133) and (134) respectively.

$$A_S = W \cdot \beta' \tag{133}$$

$$P_S = 2 \cdot \{W + NF \cdot \beta'\} \tag{134}$$

where the constant β' is given by:

$$\beta' = spcContGate + szCont + ovActCont \tag{135}$$

Since an n-well process was assumed, the PMOS transistors lie in a well. In the configuration at hand, the well is tied to the source. Because the sources may be at different potentials, one contiguous well is not possible. Rather, the n-well must be divided into *NF*/2 separate pieces. The variable K given in (124) is an expression for the width of each n-well piece. Since there are *NF*/2 different pieces, the equivalent total width of the n-well is:

$$W_{well} = \frac{NF \cdot \mathrm{K}}{2} \tag{136}$$

The height of each n-well piece is given by ().

$$H_{well} = \frac{W}{NF} + spcPActNAct + szCont + 2 \cdot ovActCont + 2 \cdot spcSWContWell \tag{137}$$

The corresponding well perimeter and area is calculated by (138) and (139) respectively.

$$P_{well} = 2 \cdot \left(W_{well} + H_{well}\right) \tag{138}$$

$$A_{well} = W_{well} H_{well} \tag{139}$$

The minimum transistor width is given by:

$$W_{\min} = NF \cdot actMinWid \tag{140}$$

The value of the constant *actMinWid* is the minimum width of an active region. Its value is specified in Table 39. Constraints are added to the design specification file to ensure that the minimum width requirements are met for each transistor.

Negative Feedback

Fig. 53 shows the block-diagram of an amplifier used in a standard feedback configuration. Assuming the loading effect of the feedback network is negligible, the closed-loop transfer function is given by (141).

$$H_{cl}(s) = \frac{v_o(s)}{v_i(s)} = \frac{H_{ol}(s)}{1 + \beta \cdot H_{ol}(s)} \tag{141}$$

Where $H_{ol}(s)$ is the open-loop transfer function of the amplifier block and β is the feedback ratio.

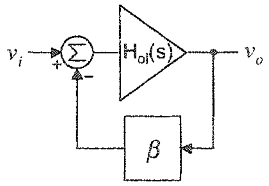


Fig. 53 Standard feedback configuration

The open-loop transfer function of can be decomposed into its numerator and denominator polynomials as:

$$H_{ol}(s) = \frac{N_A(s)}{D_A(s)} \tag{142}$$

Substituting (142) into (141) yields:

$$H_{cl}(s) = \frac{N_A/D_A}{1 + \beta(N_A/D_A)} = \frac{N_A}{D_A + \beta N_A} \tag{143}$$

Equation (143) shows how the closed-loop response is computed using the open-loop response. Note that the numerators of the open- and closed-loop transfer functions are identical. Thus, the locations of the zeros are unaffected by feedback. The denominators are, however, a different story.

A Standard Single-Stage Differential Amplifier

In this section, the development of a DSE design specification file for the fully differential amplifier depicted in Fig. (54) is described.

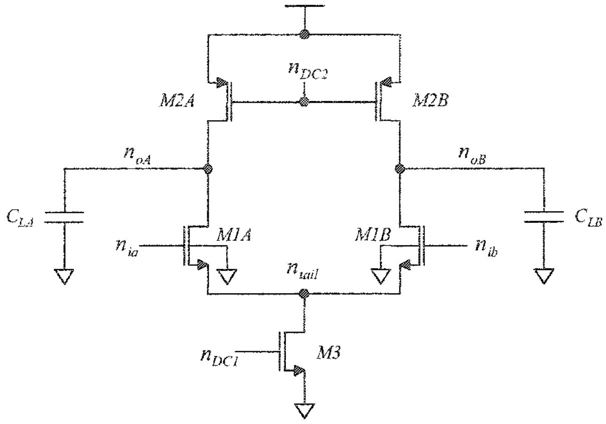


Fig. 54 Schematic diagram of a fully differential amplifier

Degrees of Freedom

The design task involves determining the physical dimensions of the devices that maximize a performance goal subject to a set of constraints. Although the physical dimensions of the devices must be determined at some point, it is not necessary to partition and explore the design-space in terms of those variables. It is often more convenient to use an alternate set of variables that result in simpler, more tractable expressions for the performance parameters [45]. For this reason, the excess biases of the devices were used as the degrees of freedom (DOFs) rather than the device widths. Table 40 lists the DOFs that were used for this design.

Table 40 Degrees of freedom used to characterize the standard fully-differential amplifier

V_{EB1}, V_{EB2}, V_{EB3}	Excess biases of devices M1-M3 (V)
L_1, L_2, L_3	Lengths of devices M1-M3 (m)
P	Total power consumption (W)
V_{DD}, V_{SS}	Supply voltages (V)
V_{ICM}	Common-mode input voltage (V)
C_L	Capacitive load (F)
DAR1, DAR2, DAR3	Desired aspect ratios of devices M1-M3
BETA	Feedback factor, for closed-loop configuration
h	Settling accuracy parameter

Common-Mode Output Voltage

The upper limit on the common-mode voltage is imposed by the condition that M2A and M2B remain in saturation.

$$V_{OCM(\max)} = V_{DD} - V_{EB2} \tag{144}$$

Likewise, the minimum common-mode output voltage is determined by the point where M1A and M1B leave saturation.

$$V_{OCM(\min)} = V_{icm} - V_{T1} \tag{145}$$

To ensure that only viable designs are considered, the following constraint was added the design specification file.

$$V_{OCM(\min)} \leq V_{OCM} \leq V_{OCM(\max)} \tag{146}$$

The choice of common-mode output voltage affects the magnitude of the maximum undistorted differential output signal that the amplifier can produce. To ensure a large differential output signal swing capability, the common-mode output voltage was chosen to be at the middle of the acceptable range.

$$V_{OCM} = \frac{V_{OCM(\min)} + V_{OCM(\max)}}{2} \tag{147}$$

Maximum Differential Output Swing

When excited differentially, the large-signal voltages n_{oA} and n_{oB} deviate differentially from the common-mode value. The maximum differential signal swing is defined as the largest signal that can be produced at the output for which all devices remain in saturation. The relationships are illustrated in Fig. 55. For a differential input of zero volts, the tail current splits evenly between the two branches and the both outputs rest at the common-mode output voltage. As the magnitude of the differential input voltage increases, more current is steered either to the left or to the right. As a result, v(n_{oA}) or v(n_{oB}) increases while the other decreases. At some point, the magnitudes of the deviations become so large that one or more of the transistors leaves saturation.

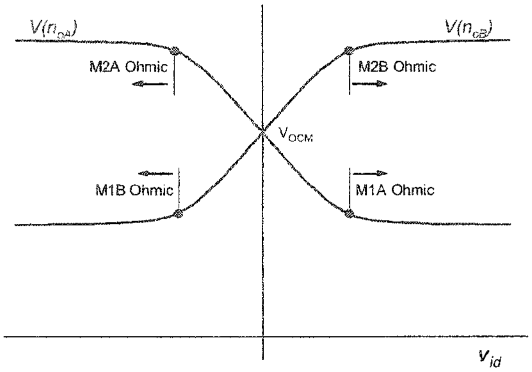


Fig. 55 Large-signal output node voltages as a function of the differential input voltage

Requiring that M2A and M2B operate in saturation limits the maximum voltages at the output nodes. Therefore, the outputs must satisfy:

$$V(n_{oA}), V(n_{oB}) \leq V_{DD} - V_{EB2} \tag{148}$$

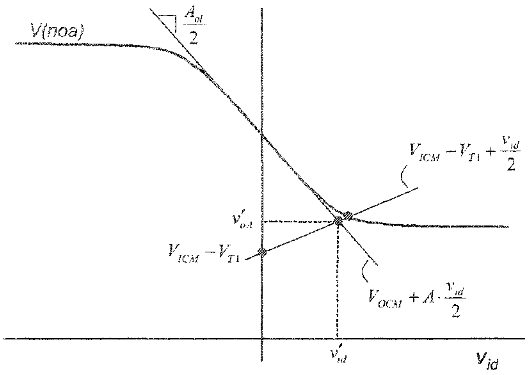


Fig. 56 Approximating the point where M1A leaves saturation

The points where M1A and M1B leave saturation correspond to the minimum nodal output voltage. Using the large-signal expressions and solving for the exact points where M1A and M1B leave saturation yields analytical expressions that are very complicated. Although coding complex analytical expressions is possible in Design Space Explorer, in this case it is not necessary. Knowing the *exact* value of the minimum nodal output voltage is not required. Errors as large as a few percent of the total output voltage swing are acceptable. For this a reason, a first-order approximation of the output voltage is employed. Fig. 56 illustrates the technique used to estimate the point where M1A leaves saturation.

M1A operates in saturation when:

$$V(n_{oA}) \geq V_{ia} - V_{T1} \tag{149}$$

For differential excitation, V_{ia} is given by $V_{ICM} + v_{id}/2$. Therefore, (149) can be written:

$$V(n_{oA}) \geq V_{ICM} + \frac{v_{id}}{2} - V_{T1} \tag{150}$$

The right-hand side of (150) is the equation of the positive-sloped line drawn in Fig. 56. The point where the line intersects the $V(n_{oA})$ curve corresponds to the minimum output voltage for differential excitation. This is the quantity that we would like to determine. However, since it is not necessary to know the exact value of the minimum output voltage and solving for the exact point requires a lot of work, the nearby point (v'_{id} , v'_{oA}) is used instead.

The approximation involves finding the point where the right-hand side of (150) intersects with a linear approximation of $V(n_{oA})$ denoted as $\tilde{V}(n_{oA})$. Mathematically, the approximation is given by:

$$\tilde{V}(n_{oA}) = V_{OCM} + \left(\frac{A_{ol}}{2}\right) v_{id} \tag{151}$$

where A_{ol} is a negative quantity known as the open-loop DC gain. Solving for the output voltage at the point of intersection.

$$v'_{oA} = \frac{V_{OCM} + |A_{ol}| \cdot (V_{ICM} - V_{T1})}{1 + |A_{ol}|} \tag{152}$$

From (148), the maximum positive deviation from the quiescent point is:

$$\Delta V_{o+} = V_{DD} - V_{EB2} - V_{OCM} \tag{153}$$

Similarly from (152), the maximum negative deviation is:

$$\Delta V_{o-} = \left(\frac{|A_{ol}|}{1 + |A_{ol}|}\right)(V_{OCM} - V_{ICM} + V_{T1}) \tag{154}$$

Since differential swings at the output should be symmetric about the common-mode output voltage, the peak-peak magnitude of the largest symmetric signal that can be accommodated on either n_{oA} or n_{oB} without clipping is given by:

$$\Delta V_{o} = 2 \cdot \min(\Delta V_{o+}, \Delta V_{o-}) \tag{155}$$

Equation (155) corresponds to the maximum single-ended swing at n_{oA} or n_{oB}. The maximum peak-peak differential swing is twice as large.

$$V_{od(\max)} = 4 \cdot \min\left(V_{DD} - V_{EB2} - V_{OCM}, \left(\frac{|A_{ol}|}{1 + |A_{ol}|}\right)(V_{OCM} - V_{ICM} - V_{T1})\right) \tag{156}$$

Notice that the common-mode output voltage affects the magnitude of the maximum undistorted differential output signal that the amplifier can produce. To maximize the signal swing, the common mode output voltage should be chosen to ensure that the maximum positive and negative deviations are equal in magnitude. Thus, the optimal common-mode output voltage is given by:

$$V_{OCM(opt)} = \frac{V_{DD} - V_{EB2} + |A_{ol}|(V_{DD} - V_{EB2} + V_{ICM} - V_{T1})}{1 + 2 \cdot |A_{ol}|} \tag{157}$$

Substituting from (144) and (145),

$$V_{OCM(opt)} = \frac{V_{OCM(\max)} + \left|A_{ol}\right| \cdot \left(V_{OCM(\max)} - V_{OCM(\min)}\right)}{1 + 2 \cdot \left|A_{ol}\right|} \tag{158}$$

Observe that, as the amplifier gain approaches infinity, the optimal value of the common-mode voltage approaches the middle of the common-mode range given in (146). For low-gain amplifiers, however, the common-mode output needs to be set a little higher.

Operating Point

The operating point is the set of node voltages and branch currents that occur in the amplifier under quiescent input conditions. For fully differential amplifiers, such as the one under consideration, a quiescent input refers to the fact the differential input voltage is zero ($V_{IA}=V_{IB}=V_{ICM}$). Since the circuit is symmetric, if both halves of the circuit match exactly, then the differential output voltage will be zero as well ($V_{OA}=V_{OB}=V_{OCM}$).

The tail current is computed by:

$$I_{DS3} = \frac{P}{V_{DD} - V_{SS}} \tag{159}$$

Assuming the differential input voltage is zero and the circuit is perfectly symmetric, the tail current splits evenly between the two branches.

$$I_{DS1} = I_{SD2} = \frac{I_{DS3}}{2} \tag{160}$$

From (56):

$$V_{EB1} = V_{GS1} - V_{T1} \tag{161}$$

Since the gate voltage is equal to the common-mode input voltage and the source is connected to the tail node, (161) can be rewritten as:

$$V_{tail} = V_{ICM} - V_{EB1} - V_{T1} \tag{162}$$

Tables 57 and 42 summarize the DC operating point currents and voltages respectively.

Table 41 Operating point currents

Current	*Value*	*Description*
I_{DS3}	$\dfrac{P}{V_{DD} - V_{SS}}$	M3 drain to source current (Tail Current)
I_{DS1}, I_{SD2}	$\dfrac{I_{DS3}}{2}$	M1 drain to source and M2 source to drain currents

Table 42 Operating point voltages

Quantity	*Node(s)*	*Value*	*Description*
V_o	n_{oA}, n_{oB}	V_{OCM}	Output
V_i	n_{iA}, n_{iB}	V_{ICM}	Input
V_{tail}	n_{tail}	$V_{ICM} - V_{EB1} - V_{T1}$	Tail
V_{DC1}	n_{DC1}	V_{DC1}	Bias
V_{DC2}	n_{DC2}	V_{DC2}	Bias

Other quantities related to the operating point are sometimes of interest to the user. Therefore, the variables shown in Table 43 have been defined in the design specification file. Using these quantities to calculate other parameters and in constraints can reduce the computational complexity of the model (see 0).

Table 43 Other operating point parameters that are defined

Quantity	Value	Description
V_{GS1}	$V_{EB1}+V_{T1}$	M1 gate-source voltage
V_{SG2}	$V_{EB2}+V_{T2}$	M2 source-gate voltage
V_{GS3}	$V_{EB3}+V_{T3}$	M3 gate-source voltage
V_{DS1}	V_o-V_{tail}	M1 drain-source voltage
V_{SD2}	$V_{DD}-V_o$	M2 source-drain voltage
V_{DS3}	$V_{tail}-V_{SS}$	M3 drain-source voltage

Device Widths

In order to physically design or even simulate the fully differential amplifier, the geometrical parameters of the devices need to be determined. However, the device widths are not present in the list of DOFs. Expressions that relate the DOFs to the device widths can be found by solving equations (54) and (55) for *W*.

The resultant expressions for device widths are given in (163) through (165).

$$W_1=\frac{I_{DS1}L_1}{\beta_n V_{EB1}^2\left(1+\lambda_n V_{DS1}\right)} \tag{163}$$

$$W_2=\frac{I_{SD2}L_2}{\beta_p V_{EB2}^2\left(1+\lambda_p V_{SD2}\right)} \tag{164}$$

$$W_3=\frac{I_{DS3}L_3}{\beta_n V_{EB3}^2\left(1+\lambda_n V_{DS3}\right)} \tag{165}$$

Small-Signal Model

Fig. 57 shows the complete small-signal model of the fully-differential amplifier shown in Fig. 54.

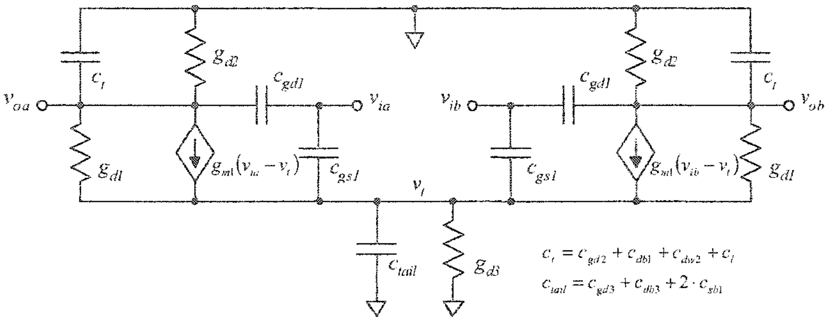


Fig. 57 Small signal model of the fully-differential amplifier

Since the amplifier is symmetric, its response will also be symmetric. Therefore, using the procedure described in 0, the total small-signal response can be determined if you know the differential and common-mode responses. For differential excitation, the tail node acts as a virtual ground and the small-signal model of Fig. 57 simplifies to the model shown in Fig. 58. The output conductance, g_{dd}, and total output node capacitance, c_t, are given by (166) and (167) respectively.

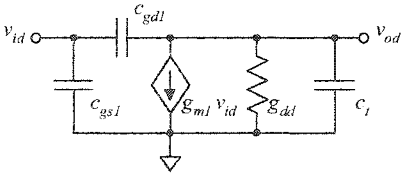


Fig. 58 Simplified small signal model for differential excitation

$$g_{dd} = g_{d1} + g_{d2} \tag{166}$$

$$c_t = c_{gd2} + c_{db1} + c_{dw2} + c_l \tag{167}$$

Analyzing the circuit in Fig. 58 reveals that the differential transfer function is:

$$H_d(s) = \frac{-(g_{m1} - s \cdot c_{gd1})}{g_{dd} + s \cdot (c_{gd1} + c_t)} \tag{168}$$

When using a fully differential amplifier, differential signaling is normally employed. The signal to be amplified is encoded as the difference between two signals whose common-mode value is held constant. For this reason, a designer is primarily interested in the differential-mode transfer function and the common-mode transfer function is not of interest. In fact, a special circuit called a common-mode feedback circuit is usually included to stabilize the common-mode value of the output signals.

The small-signal output conductances are determined using (63) and (64).

$$g_{d1} = \frac{\lambda_n I_{DS1}}{1+\lambda_n V_{DS1}} \tag{169}$$

$$g_{d2} = \frac{\lambda_p I_{SD2}}{1+\lambda_p V_{SD2}} \tag{170}$$

M1's small-signal transconductance is computed using (61).

$$g_{m1} = \frac{2I_{DS1}}{V_{EB1}} \tag{171}$$

In order to accurately estimate the drain-well and drain-source capacitances, knowledge of the layout style is required. Since the transistor layouts are not available, some were assumed. An effort was made to choose appropriate layout styles. However, if they are not suitable for a specific application the diffusion area and perimeter calculations will need to be modified. M1A and M1B are assumed to be laid-out using an interdigitized common-centroid layout like the one described in 0. Similarly, M2A and M2B are assumed to be laid-out using an interdigitized common-centroid layout like the one described in 0. Finally, M3 is laid-out using the fingered layout described in 0. Tables 44 and 45 show the calculations used to compute the diffusion perimeters and areas. Table 46 shows the calculations used to determine the number of fingers in the transistor layouts that most closely achieve the desired aspect ratios.

Table 44 Diffusion perimeter calculations

Name	Diffusion	Perimeter
P_{D1}	D1A, D1B	$W_1 + NF_1 \cdot \beta$
P_{S1}	S1	$\left(\frac{NF_1+1}{NF_1}\right)P_{D1}$
P_{D2}	D2A, D2B	$W_2 + NF_2 \cdot \beta$
P_{D3}	D3	$W_3 + NF_3 \cdot \beta$

Table 45 Diffusion area calculations

Name	Diffusion	Area
A_{D1}	D1A, D1B	$\frac{W_1 \cdot \beta}{2}$
A_{S1}	S1	$\left(\frac{NF_1 + 1}{NF_1}\right) A_{D1}$
A_{D2}	D2A, D2B	$\frac{W_2 \cdot \beta}{2}$
A_{D3}	D3	$\frac{W_3 \cdot \beta}{2}$

Table 46 Calculations to determine the number of fingers/transistor

Name	Calculation
β	$szCont + 2 \cdot spcContGate$
α_k	$\frac{\beta}{4(L_k + \beta)}, \qquad k \in \{1,2,3,4,5\}$
NF_1	$\max\left(4 \cdot \text{rint}\left(\alpha_1 \cdot \left(\sqrt{1 + \frac{2 \cdot W_1 \cdot DAR_1}{\alpha_1 \cdot \beta}} - 1\right) \Big/ 4\right), 4\right)$
NF_2	$\max\left(4 \cdot \text{rint}\left(\alpha_2 \cdot \left(\sqrt{1 + \frac{2 \cdot W_2 \cdot DAR_2}{\alpha_2 \cdot \beta}} - 1\right) \Big/ 4\right), 4\right)$
NF_3	$\max\left(2 \cdot \text{rint}\left(2 \cdot \alpha_3 \cdot \left(\sqrt{1 + \frac{W_3 \cdot DAR_3}{\alpha_3 \cdot \beta}} - 1\right) \Big/ 2\right), 2\right)$

The procedure used to estimate the parasitic capacitances of a FET operating in saturation is described in section 0. Table 47 details the equations used to compute the sizes of the relevant parasitic elements. The parasitic elements that have fixed DC potentials across them have been omitted. They are shorted in the small-signal model and therefore can be neglected.

Table 47 Parasitic capacitance calculations

Name	Capacitance Calculation
C_{gd1}	$C_{oxn}W_1L_D$
C_{gd2}	$C_{oxp}W_2L_D$
C_{gd3}	$C_{oxn}W_3L_D$
C_{gs1}	$C_{oxn}W_1L_D + \frac{2}{3}C_{oxn}W_1L_1$
C_{dw2}	$\frac{C_{JP}A_{D2}}{\left(1+\frac{V_{DD}-V_o}{\phi_{BP}}\right)^{MJP}} + \frac{C_{JSWP}P_{D2}}{\left(1+\frac{V_{DD}-V_o}{\phi_{BP}}\right)^{MJSWP}}$
C_{db1}	$\frac{C_{JN}A_{D1}}{\left(1+\frac{V_o-V_{SS}}{\phi_{BN}}\right)^{MJN}} + \frac{C_{JSWN}P_{D1}}{\left(1+\frac{V_o-V_{SS}}{\phi_{BN}}\right)^{MJSWN}}$
C_{db3}	$\frac{C_{JN}A_{D3}}{\left(1+\frac{V_{tail}-V_{SS}}{\phi_{BN}}\right)^{MJN}} + \frac{C_{JSWN}P_{D3}}{\left(1+\frac{V_{tail}-V_{SS}}{\phi_{BN}}\right)^{MJSWN}}$
C_{sb1}	$\frac{C_{JN}A_{S1}}{\left(1+\frac{V_{tail}-V_{SS}}{\phi_{BN}}\right)^{MJN}} + \frac{C_{JSWN}P_{S1}}{\left(1+\frac{V_{tail}-V_{SS}}{\phi_{BN}}\right)^{MJSWN}}$

Open-Loop Response

The small-signal model of the differential amplifier shown in Fig. 58 has an open-loop transfer function given by (168). It is rewritten here in a slightly different form.

$$H_{ol}(s) = \frac{A_{ol}\left(1 - \frac{s}{Z}\right)}{\left(1 - \frac{s}{P_{ol}}\right)} \tag{172}$$

The quantities A_{ol}, P_{ol} and Z represent the open-loop DC gain, open-loop pole location, and zero location. The expressions used to compute these quantities are given in (173), (174), and (175).

$$A_{ol} = -\frac{g_{m1}}{g_{dd}} \tag{173}$$

$$P_{ol} = \frac{-g_{dd}}{c_t + c_{gd1}} \tag{174}$$

$$Z = \frac{g_{m1}}{c_{gd1}} \tag{175}$$

The values of the variables g_{dd} and c_t that appear in (173)-(175) are given in (166) and (167). Typically, $c_t >> c_{gd1}$. Therefore, the zero normally lies at frequencies that are well beyond the unity-gain frequency of the amplifier. As a result, the zero has a negligible effect over the frequency band of interest and is normally neglected without a significant loss of accuracy. Another parameter that is frequently of interest to designers is the unity-gain frequency. An expression for the unity gain frequency is given in (176).

$$\omega_u = \sqrt{\frac{(g_{m1} - g_{dd})(g_{m1} + g_{dd})}{c_t(c_t + 2 \cdot c_{gd1})}} \approx \frac{g_{m1}}{c_t} \tag{176}$$

Since designers are accustomed to using Bode plots, plot elements were added to the design specification file to produce them.

Closed-Loop Response

The closed-loop response is computed from the open-loop response via (143).

$$H_{cl}(s) = \frac{A_{cl}\left(1-\dfrac{s}{Z}\right)}{\left(1-\dfrac{s}{P_{cl}}\right)} \tag{177}$$

The quantities A_{cl}, P_{cl} and Z represent the closed-loop DC gain, closed-loop pole location, and zero location. The expressions used to compute these quantities are given in (178), (179), and (180).

$$A_{cl} = \frac{A_{ol}}{1+A_{ol}\beta} \tag{178}$$

$$P_{cl} = \frac{Z \cdot P_{ol} \cdot (1 + A_{ol} \cdot \beta)}{(Z + A_{ol} \cdot \beta \cdot P_{ol})} \approx P_{ol} \cdot (1 + A_{ol} \cdot \beta) \tag{179}$$

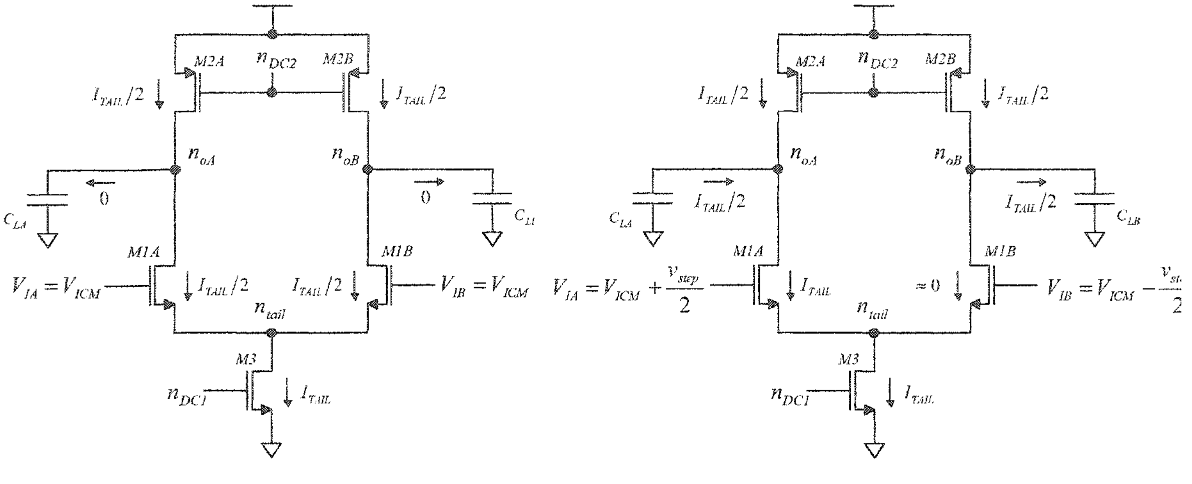
$$Z = \frac{g_{m1}}{c_{gd1}} \tag{180}$$

Note that the closed-loop transfer function has the same functional form as the open-loop transfer function. However, depending on the value of β, their pole locations and DC gains may differ. The location of the zero is unaffected by feedback. As the case was for the open-loop response, if the zero lies at frequencies well beyond the unity-gain frequency of the amplifier, the zero can be neglected without incurring a significant loss of accuracy.

Slew Rate

The maximum rate at which the output voltage can change is determined by the amount of current available to charge/discharge the capacitive loads. To illustrate, consider the quiescent amplifier shown in Fig. 59(a). Assuming perfect matching, the tail current splits evenly between the left and right circuit halves so that the quiescent current through M1A, M1B, M2A, and M2B are all equal to $I_{TAIL}/2$.

Suppose that a large, instantaneous step-change is applied to the inputs of the amplifier. The values of the currents at the instant immediately following the input change are labeled on the diagram shown in Fig. 59(b). It was assumed that the input step was of sufficient magnitude to steer virtually all the tail current to the left. Therefore, at the instant after the input step, C_{LA} is being discharged and C_{LB} is being charged by currents equal to I_{TAIL}/2. Since all of the current is being steered either to the right or to the left, these charge/discharge rates represent the maximum rate of output change possible for the architecture.



(a)

(b)

Fig. 59 Branch currents (a) under quiescent conditions (b) immediately after a large input step-change

The current-voltage relationship at the terminals of a capacitor is given by:

$$I = c\frac{dV}{dt} \tag{181}$$

The total capacitance on each of the output nodes evaluated at the operating point is given by:

$$c_t = c_{gd2} + c_{db1} + c_{dw2} + c_l \tag{182}$$

Therefore the maximum rates at which V_{oA} and V_{oB} change are given by (183) and (184).

$$\frac{dV_{oA}}{dt} = \frac{-I_{TAIL}}{2 \cdot c_t} \tag{183}$$

$$\frac{dV_{oB}}{dt} = \frac{I_{TAIL}}{2 \cdot c_t} \tag{184}$$

Since the differential output voltage, V_{od}, is the difference in voltage between V_{oB} and V_{oA}, the maximum rate of change in V_{od} is given by:

$$\frac{dV_{od}}{dt}=\frac{dV_{oB}}{dt}-\frac{dV_{oA}}{dt} \tag{185}$$

Substituting (183) and (184) into (185) results in the absolute maximum rate of change in the differential output voltage. This quantity is referred to as the *slew-rate* (SR).

$$SR=\frac{I_{tail}}{C_l} \tag{186}$$

Settling Time

The step-response settling time is a measure of how fast an amplifier responds to step changes at its inputs. Specifically, the term refers to the minimum amount of time that must elapse after a step is applied until it can be guaranteed that the present and all future values of the output will lie within a certain tolerance of a specific target output value. The process is illustrated in Fig. 60. Settling-time is an important parameter because it can be used to determine the maximum speed of operation of many circuits

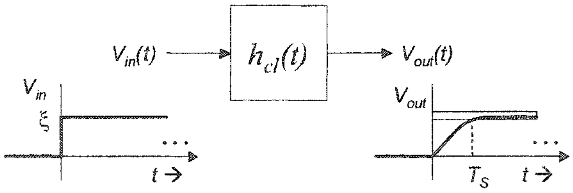


Fig. 60 Step response settling time block diagram

Depending upon the application, settling time is usually defined in one of two ways. Fig. 61 illustrates the two methods. All points in the shaded regions correspond to signal values that are close enough to the specified value to be considered *settled.* These regions are referred to as *settling windows*. In this case, the width of the settling window (the settling accuracy) is parameterized by the variable *h*. Smaller values of h correspond to greater settling accuracies. For example, for 1% settling accuracy, *h* would be 0.01.

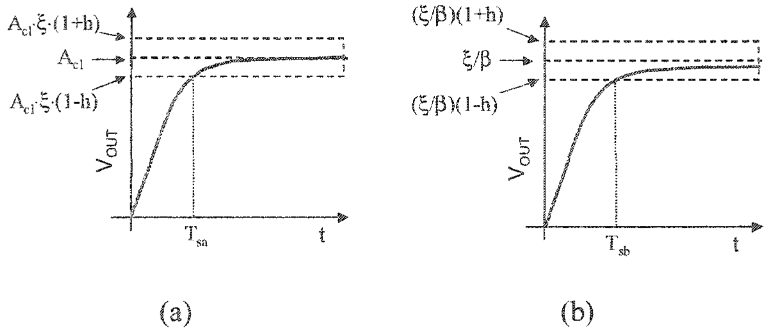


Fig. 61 Two commonly used definitions of settling time (a) relative to the asymptotic steady-state value, (b) relative to the desired value of 1/β

In Fig. 61(a), the center of settling region is defined relative to the asymptotic steady-state value of the waveform. This definition of settling time is often preferred in mechanics and other control systems. It might, for example, refer to the amount of time it takes for an object's physical vibrations to diminish to a certain level after it has been jarred. Alternatively, as shown in Fig. 61(b), the settling window can be defined relative to a specific value (in this case, 1/β). Data converter designers sometimes prefer this definition because their application may require a gain of exactly 2, for example.

Assuming the closed-loop system of (177) is initially at rest, the transient response to an ideal step applied at $t=0$ can be written as shown in (187).

$$v_{od}(t) = v_{od}(0^-) + A_{cl}\cdot\xi\cdot\left[1-\left(1-\frac{P_{cl}}{Z}\right)e^{P_{cl}\cdot t}\right]u(t) \qquad (187)$$

The variable ξ represents the signed magnitude of the input step and $v_{od}(0^-)$ is the value of the differential output voltage at time zero approached from the left. The response of (187) is sketched in Fig. 62. Due to the instantaneous step-change in the input at $t=0$ and the frequency-independent response of the capacitive divider formed by c_{gd} and c_t in the small-signal model, except in trivial cases, $v_{od}(t)$ is discontinuous at $t=0$. In most instances, though, the magnitude of the discontinuity is modest and is safely neglected.

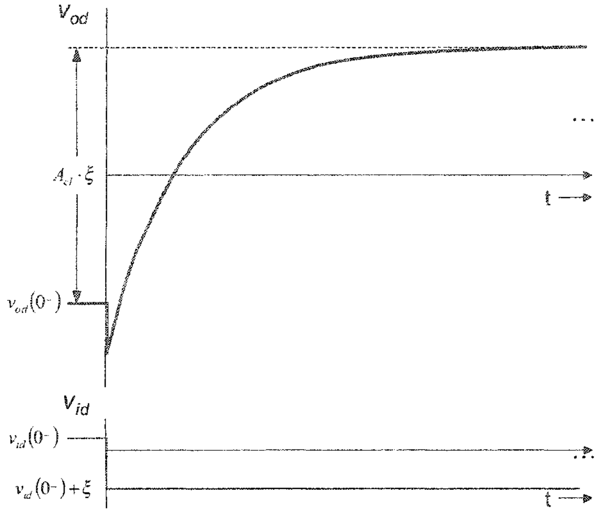


Fig. 62 Linear step-response of a differential amplifier

The required rate of output change that the circuit must be able to provide can be determined by differentiating (187).

$$\frac{dv_{od}}{dt} = -A_{cl}\cdot\xi\cdot P_{cl}\cdot\left(1-\frac{P_{cl}}{Z}\right)\cdot e^{P_{cl}\cdot t}\cdot u(t) \tag{188}$$

For stable systems, P_{cl} is in the left half-plane. As a result, $\frac{dv_{od}}{dt}$ decays with time. Therefore, the absolute maximum rate of change occurs at $t=0^+$. Define φ as the magnitude of (188) evaluated at $t=0$.

$$\varphi = \left|\lim_{t\to 0^+}\left(\frac{dv_{od}}{dt}\right)\right| = \left|A_{cl}\cdot\xi\cdot P_{cl}\cdot\left(1-\frac{P_{cl}}{Z}\right)\right| \tag{189}$$

If φ exceeds the slew-rate capability of the amplifier (SR), the amp will initially slew toward its final value. Thus, the magnitude of the largest input step that can be processed without slewing is given by:

$$\xi_{\max} = \frac{SR}{A_{cl}\cdot P_{cl}\cdot\left(1-\frac{P_{cl}}{Z}\right)} \tag{190}$$

For step-changes larger in magnitude than ξ_{max}, the amplifier will initially slew toward its final output at a constant rate. As the output approaches the final value, the rate of output change required by the linear model diminishes. Slewing continues until the rate of output change required by the linear model and the slew-rate capability of the amplifier are equal. Denote this point in time as T_{sl}. From time T_{sl} on, the amplifier settles linearly. As illustrated in Fig. 63, neglecting slew-rate effects might result in significant under-estimation of the actual settling time (*Error* = $T_2 - T_1$).

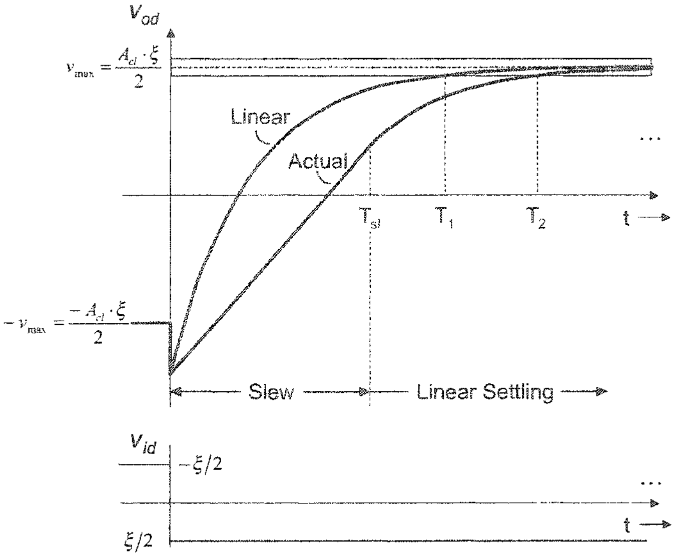


Fig. 63 Diagram illustrating how slew-rate limiting can grossly affect the settling time

Assume we have a stable system excited by a step-input that is larger in magnitude than ξ_{max}. During the interval $0 \le t \le T_{sl}$, the output slews toward its final value at a constant rate according to the relationship given in (191).

$$v_{od}(t) = v_{od}(0^+) - \frac{\xi}{|\xi|}\cdot SR\cdot t \qquad\qquad 0 \le t \le T_{sl} \qquad (191)$$

In (191), $v_{od}(0^+)$ refers to the output voltage immediately after the input transition. This formalism is necessary to account for the discontinuity that can appear in the output voltage due to an instantaneous change in the input voltage. The value of $v_{od}(0^+)$ is determined by evaluating (187) at $t=0$ and is given in (192).

$$v_{od}(0^+) = v_{od}(0^-) + \frac{A_{cl}\cdot P_{cl}}{Z}\xi \qquad (192)$$

As the output approaches the final value, the rate of output change required to support linear settling diminishes. At time T_{sl}, the system resumes linear operation. The subsequent linear response can be expressed as:

$$v_{od}(t) = v_{od}(\infty) - [v_{od}(\infty) - v_{od}(T_{sl})] \cdot e^{P_{cl} \cdot (t - T_{sl})} \qquad t \geq T_{sl} \tag{193}$$

The coefficient of the exponential term is the magnitude of the transient component at time $t=T_{sl}$. $v_{od}(\infty)$ represents the asymptotic steady-state value of the output while $v_{od}(T_{sl})$ is the value of the output when the circuit stops slewing. An expression for $v_{od}(\infty)$ is obtained by evaluating the limit of (187) as t approaches infinity.

$$v_{od}(\infty) = v_{od}(0^-) + A_{cl} \cdot \xi \tag{194}$$

$v_{od}(T_{sl})$ is determined by substituting (192) into (191) and evaluating at time $t=T_{sl}$.

$$v_{od}(T_{sl}) = v_{od}(0^-) + \frac{A_{cl} \cdot P_{cl}}{Z} \cdot \xi - \frac{\xi}{|\xi|} \cdot SR \cdot T_{sl} \tag{195}$$

Substituting (194) and (195) into (193) and simplifying yields (196) which is only valid for $t \geq T_{sl}$.

$$v_{od}(t) = v_{od}(0^-) + A_{cl} \cdot \xi - \left\{ A_{cl} \cdot \xi \cdot \left(1 - \frac{P_{cl}}{Z}\right) + \frac{\xi}{|\xi|} \cdot SR \cdot T_{sl} \right\} e^{P_{cl} \cdot (t - T_{sl})} \tag{196}$$

Slewing ceases and linear settling resumes as soon as the amplifier is able to support the slew requirements of the linear model. Differentiating (196) with respect to t yields the rate of output change that the amplifier must be able to support.

$$\frac{dv_{od}}{dt} = -P_{cl} \cdot \left\{ A_{cl} \cdot \xi \cdot \left(1 - \frac{P_{cl}}{Z}\right) + \frac{\xi}{|\xi|} \cdot SR \cdot T_{sl} \right\} e^{P_{cl} \cdot (t - T_{sl})} \qquad t \geq T_{sl} \tag{197}$$

For stable systems, P_{cl} is in the left half-plane. Therefore, (197) decays with time and the absolute maximum slew-rate occurs at $t=T_{sl}$.

$$\left. \frac{dv_{od}}{dt} \right|_{t=T_{sl}} = -P_{cl} \cdot \xi \cdot \left\{ A_{cl} \cdot \left(1 - \frac{P_{cl}}{Z}\right) + \frac{SR}{|\xi|} \cdot T_{sl} \right\} \tag{198}$$

To determine T_{sl}, equate (198) to the slew-rate capability of the amplifier.

$$-P_{cl} \cdot \xi \cdot \left\{ A_{cl} \cdot \left(1 - \frac{P_{cl}}{Z}\right) + \frac{SR}{|\xi|} \cdot T_{sl} \right\} = \frac{\xi}{|\xi|} \cdot SR \tag{199}$$

Solving (199) for T_{sl} and designating the result as T'_{sl} yields (200).

$$T'_{sl} = \left(\frac{-1}{P_{cl}}\right) \cdot \left(\frac{|\xi|}{\xi_{\max}} - 1\right) \tag{200}$$

ξ_{max}, given in (190), is the magnitude of the largest input step that can be applied to the system that does not result in slewing. Thus, the rightmost parenthetical term is always greater than zero for systems that slew and always less than zero for those that do not slew. For stable closed-loop systems, the leftmost parenthetical term is always greater than zero. Therefore, if T'_{sl} is positive, the system slews for T'_{sl} seconds. If T'_{sl} is negative, no slewing occurs.

Since no slewing occurs in cases where T'_{sl} is negative and negative slew-time is not sensible, a unit-step function is used to eliminate the negative values. The final expression for the worst-case slew time is given in (204).

$$T_{sl} = T'_{sl} \cdot u(T'_{sl}) \tag{201}$$

In terms of settling, full-scale output transitions are the most challenging. The maximum peak-peak output swing is given by $V_{od(max)}$ defined in (156). Since the closed-loop gain is A_{cl}, the magnitude of the input step corresponding to a full-scale output swing is given by:

$$\xi_{wc} = \frac{\mathrm{V}_{\mathrm{od(max)}}}{|A_{cl}|} \tag{202}$$

Therefore, the worst-case slew time is computed by substituting (202) into (200).

$$T'_{sl\_wc} = \left(\frac{-1}{P_{cl}}\right) \cdot \left(\frac{\mathrm{V}_{\mathrm{od(max)}}}{|\mathrm{A}_{\mathrm{cl}}| \cdot \xi_{\max}} - 1\right) \tag{203}$$

Substituting (203) into (201) yields the final expression for the worst-case slew time.

$$T_{sl\_wc} = T'_{sl\_wc} \cdot u(T'_{sl\_wc}) \tag{204}$$

Equation (204) is an expression for the total amount of time spent slewing after a step-change in the input. To determine the total time required for settling, the time required to settle from the post-slew value to within a certain tolerance of the desired value must be computed.

The settling time differs based upon which definition of settling time is employed. The first method which is depicted in Fig. 61(a) will be referred to as method *A*. It defines the settling band relative to the asymptotic steady-state value of the output waveform. Since the response does not ring or overshoot, the settling time, T_{sa}, is the point in time where the output voltage given by (196), intersects the inner edge of the settling band. Mathematically this is written as:

$$v_{od}(0^-)+A_{cl}\cdot\xi-\left\{A_{cl}\cdot\xi\cdot\left(1-\frac{P_{cl}}{Z}\right)+\frac{\xi}{|\xi|}\cdot SR\cdot T_{sl}\right\}e^{P_{cl}\cdot(T_{sa}-T_{sl})}=v_{od}(0^-)+A_{cl}\cdot\xi\cdot(1-h) \qquad (205)$$

Solving (205) for T_{sa}:

$$T_{sa}=\left(\frac{-1}{P_{cl}}\right)\cdot\ln\left(\frac{A_{cl}\cdot\left(1-\frac{P_{cl}}{Z}\right)+\frac{SR\cdot T_{sl}}{|\xi|}}{A_{cl}\cdot h}\right)+T_{sl} \qquad (206)$$

The worst-case settling time is determined by substituting (202) into (206).

$$T_{sa\_wc}=\left(\frac{-1}{P_{cl}}\right)\cdot\ln\left(\frac{\left(1-\frac{P_{cl}}{Z}\right)+\frac{SR\cdot T_{sl\_wc}}{V_{od(\max)}}}{h}\right)+T_{sl\_wc} \qquad (207)$$

The second definition of settling time defines settling relative to the desired gain of 1/β. This method is graphically depicted in Fig. 61(b) and will be referred to as method *B*. The settling time, T_{sb}, is the point in time where the output voltage intersects the inner edge of the settling band. For settling to take place, the DC gain of the amplifier must be large enough to ensure that the output waveform settles to a value that lies inside of the settling window. Therefore, for finite settling times, the following inequality must be satisfied:

$$(1-h)<A_{cl}\cdot\beta<(1+h) \qquad (208)$$

Settling occurs when the output given by (196) crosses the inner edge of the settling window.

$$v_{od}(0^-)+A_{cl}\cdot\xi-\left\{A_{cl}\cdot\xi\cdot\left(1-\frac{P_{cl}}{Z}\right)+\frac{\xi}{|\xi|}\cdot SR\cdot T_{sl}\right\}e^{P_{cl}\cdot(T_{sa}-T_{sl})}=v_{od}(0^-)+\frac{\xi}{\beta}\cdot(1-h) \qquad (209)$$

Solving (209) for T_{sb} and designating the result as T'_{sb}:

$$T'_{sb} = \left(\frac{-1}{P_{cl}}\right)\cdot \ln\left(\frac{A_{cl}\cdot\left(1-\frac{P_{cl}}{Z}\right)+\frac{SR\cdot T_{sl}}{|\xi|}}{A_{cl}-(1-h)}\right)+T_{sl} \tag{210}$$

Equation (210) can be rewritten as:

$$T'_{sb} = \left(\frac{-1}{P_{cl}}\right)\cdot \ln\left(\frac{A_{cl}\cdot\left(1-\frac{P_{cl}}{Z}\right)+\frac{SR\cdot T_{sl}}{|\xi|}}{A_{cl}\cdot h}\right)+T_{sl}+\left(\frac{-1}{P_{cl}}\right)\cdot \ln\left(\frac{A_{cl}\cdot\beta\cdot h}{A_{cl}\cdot\beta-(1-h)}\right) \tag{211}$$

Substituting from (206) yields:

$$T'_{sb} = T_{sa}+\left(\frac{-1}{P_{cl}}\right)\cdot \ln\left(\frac{A_{cl}\cdot\beta\cdot h}{A_{cl}\cdot\beta-(1-h)}\right) \tag{212}$$

According to (208), for systems with enough DC gain to ensure settling within the desired settling window, the denominator of (212) is always positive. One could add a constraint to the design specification file to ensure that designs that don't satisfy $A_{cl}\cdot\beta > (1-h)$ get tagged as unviable. However, not all applications define settling relative to 1/β. In some cases, designs that define settling relative to the asymptotic steady state value would be tagged as unviable even though they are viable. Therefore, rather than using a constraint, a filtering system based on the unit-step function has been employed.

$$T_{sb} = (-1)\cdot u(-A_{cl}\cdot\beta+(1-h))+(T'_{sb})\cdot u(A_{cl}\cdot\beta-(1-h)) \tag{213}$$

Equation (213) filters (212). For cases where the DC gain is inadequate to settle to a value that lies within the desired settling window, the output is assigned the value of -1. For cases with adequate gain, the value of the function is identical that of (212).

Transient response

A full-scale output transition is worst-case in terms of slewing and settling. The mathematical expressions for the worst-case transient behavior were derived in section 0. The relevant equations are repeated in (214).

$$v_{od}(t)=\begin{cases} v_{od}\left(0^-\right) & t<0 \\ v_{od}\left(0^+\right)-\frac{\xi}{|\xi|}\cdot SR\cdot t & 0<t\leq T_{sl} \\ v_{od}(\infty)-\left[v_{od}(\infty)-v_{od}\left(T_{sl}\right)\right]\cdot e^{P_{cl}\cdot(t-T_{sl})} & t\geq T_{sl} \end{cases} \qquad (214)$$

Spice Netlist

A parameterized netlist has been prepared to make it easy to verify a design using WinSpice. A DSE *filter* element in the design specification file add a "Misc." menu that allows the user to trigger the creation of a customized netlist. After the netlist is created, the user should save the custom file and source it in WinSpice to simulate the performance of the amplifier.

A nested sweep automatically adjusts V_{DC2} to the value required to make the quiescent common-mode output voltage equal to the desired value. A small signal AC analysis is performed at the Q-point. The large signal quasi-static transfer characteristic is computed. Plots of the small-signal gain as a function of signal swing are produced and the full-scale transient step-response is simulated and plotted. The netlists are provided as a starting point for simulation. Feel free to modify the netlists according to your individual needs.

The Telescopic Cascode Amplifier

In this section, the development of a design specification file for the fully-differential telescopic cascode amplifier shown in Fig. 64 is described. This architecture has been very popular and widely used over the years. Its primary drawback limiting its use today is its limited output swing capability. The large number of devices stacked between the supply rails limits the available room for signal swing.

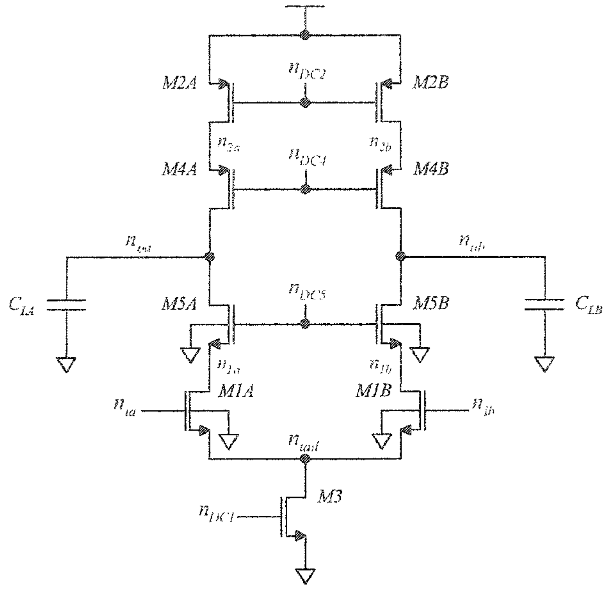


Fig. 64 Schematic diagram of the telescopic cascode amplifier

Degrees of Freedom

The degrees of freedom used for the telescopic cascode amplifier are listed in Table 48. For the same reasons described in 0, transistor excess biases are used as the degrees of freedom instead of the device widths.

Table 48 Degrees of freedom used to characterize the telescopic cascode amplifier

V_{EB1}, V_{EB2}, V_{EB3}, V_{EB4}, V_{EB5}	Excess biases of devices M1-M5 (V)
L_1, L_2, L_3, L_4, L_5	Lengths of devices M1-M5 (m)
P	Total power consumption (W)
V_{DD}, V_{SS}	Supply voltages (V)
V_{ICM}	Common-mode input voltage (V)
C_L	Capacitive load (F)
DAR1, DAR2, DAR3, DAR4, DAR5	Desired aspect ratios of devices M1-M5
BETA	Feedback factor, for closed-loop configuration
h	Settling accuracy parameter

Common-Mode Output Voltage

The common-mode range is the range of possible quiescent common-mode output voltages for which all devices operate in saturation. The upper limit on the common-mode output voltage is imposed by M4A and M4B. The maximum common-mode output voltage for which they operate in saturation is given by:

$$V_{OCM(\max)} = V_{DC4} + |V_{T4}| \tag{215}$$

Likewise, the lower limit is imposed by M5A and M5B. To operate in saturation, the common-mode output voltage must not drop below a certain level.

$$V_{OCM(\min)} = V_{DC5} - V_{T5} \tag{216}$$

To ensure only viable designs are considered, the following constraint was added to the design specification file.

$$V_{OCM(\min)} \le V_{OCM} \le V_{OCM(\max)} \tag{217}$$

The choice of common-mode output voltage affects the magnitude of the maximum undistorted differential output signal that the amplifier can produce. Since signal swing is often the factor limiting the viability of these architectures, the common-mode voltage will be chosen to maximize the differential output signal swing.

Maximum Differential Output Swing

The maximum positive voltage excursion on either of the output nodes from the common-mode voltage is given by:

$$\Delta V_{o+} = V_{DC4} + |V_{T4}| - V_{OCM} \tag{218}$$

Likewise the magnitude of the largest negative voltage excursion on either of the output nodes from the common-mode value is:

$$\Delta V_{o-} = V_{OCM} - V_{DC5} + V_{T5} \tag{219}$$

Therefore, the peak-peak magnitude of the largest symmetric signal that can be accommodated on either of the output nodes without any devices leaving saturation is:

$$\Delta V_{o} = 2 \cdot \min(\Delta V_{o+}, \Delta V_{o-}) \tag{220}$$

This corresponds to the maximum single-ended swing at n_{oa} or n_{ob}. The maximum peak-peak differential swing is twice as large.

$$V_{od(\max)} = 4 \cdot \min(\Delta V_{o+}, \Delta V_{o-}) \tag{221}$$

To maximize the output swing, the magnitudes of the peak positive and negative excursions should be balanced. Equating (218) to (219) and solving for the common-mode output voltage that will maximize the output swing results in:

$$V_{OCM} = \frac{(V_{DC5} - V_{T5}) + (V_{DC4} - V_{T4})}{2} \tag{222}$$

Substituting from (215) and (216)

$$V_{OCM} = \frac{V_{OCM(\min)} + V_{OCM(\max)}}{2} \tag{223}$$

Thus, neglecting transistor body-effect, setting the common-mode voltage exactly in the middle of its acceptable range maximizes the differential signal swing.

Operating Point

The *tail current* is determined by the total power.

$$I_{DS3} = \frac{P}{V_{DD} - V_{SS}} \tag{224}$$

Assuming the differential input voltage is zero and the circuit is perfectly symmetric, the tail current splits evenly between the two branches.

$$I_{DS1} = I_{SD2} = I_{SD4} = I_{DS5} = \frac{I_{DS3}}{2} \tag{225}$$

Equations (56) and (57) relate the transistor excess biases to their gate-source voltages.

$$V_{GS1} = V_{EB1} + V_{T1} \tag{226}$$

$$V_{SG2} = V_{EB2} - V_{T2} \tag{227}$$

$$V_{GS3} = V_{EB3} + V_{T3} \tag{228}$$

$$V_{SG4} = V_{EB4} - V_{T4} \tag{229}$$

$$V_{GS5} = V_{EB5} + V_{T5} \tag{230}$$

The DC bias voltages applied to nodes n_{DC2} and n_{DC3} are:

$$V_{DC2} = V_{DD} - V_{SG2} \tag{231}$$

$$V_{DC3} = V_{SS} + V_{GS3} \tag{232}$$

The cascode transistors M4A and M4B buffer nodes n_{2a} and n_{2b} from the output nodes. As a result, the signal variations at n_{2a} and n_{2b} are about g_{d4}/g_{m4} times the size of the output variations. The DC voltage at n_{2a} and n_{2b} should be set low enough to ensure M2A and M2B remain in saturation for full-scale output variations. Additional margin should also be added to account for the variations in V_{DC2} due to the common-mode feedback circuit compensating for process and environmental variations. Based on these factors, one might be tempted set the quiescent voltage low and not worry about it. This, however, is not a good idea because the size of the maximum signal swing is reduced in direct proportion to the amount that V_2 is lowered.

Since M2 and M4 must remain in saturation, the maximum voltage that either output, V_{oa} or V_{ob} could possibly achieve *(roughly)* is V_{DD} - V_{EB2} - V_{EB4}. Likewise, the minimum output voltage either output could achieve and still ensure M1, M3, and M5 remain in saturation is $V_{SS} + V_{EB1} + V_{EB3} + V_{EB5}$. Therefore, a crude upper bound on the maximum output node swing is:

$$\Delta V_o = (V_{DD} - V_{SS}) - (V_{EB1} + V_{EB2} + V_{EB3} + V_{EB4} + V_{EB5}) \tag{233}$$

Since the ratio of g_{m4}/g_{d4} is typically 10 or more, the peak-peak swing at n_{2a} or n_{2b} is expected to be no larger than:

$$\Delta V_2 = \frac{(V_{DD} - V_{SS}) - (V_{EB1} + V_{EB2} + V_{EB3} + V_{EB4} + V_{EB5})}{10} \tag{234}$$

V_2 is chosen to be:

$$V_2 = (V_{DD} - V_{EB2}) - \frac{\Delta V_2}{2} - V_\delta \tag{235}$$

The parenthesized term corresponds to the maximum voltage at n_{2a} and n_{2b} that still allows M2A and M2B to operate in saturation. The next term adds enough margin to ensure that the devices remain saturated in the presence of signal swings. The final term compensates for the expected process and environment variability all lumped together in one constant. The value of V_δ is not usually available but it could be estimated if good statistical process models were available. For this work, since the process data is not available, a constant value of 100mV is used instead.

Now that V_2 is known, V_{DC4} can be determined.

$$V_{DC4} = V_2 - V_{SG4} \tag{236}$$

Given the common-mode input voltage, V_{ICM}, the tail voltage can be expressed as:

$$V_{TAIL} = V_{ICM} - V_{GS1} \tag{237}$$

A similar procedure to the one used to determine the value of V_2 is employed to determine the quiescent voltage at nodes n_{1a} and n_{1b}.

$$V_1 = (V_{TAIL} + V_{EB1}) + \frac{\Delta V_1}{2} + V_\delta \tag{238}$$

Once V_1 is known, V_{DC5} can be determined.

$$V_{DC5} = V_1 + V_{GS5} \tag{239}$$

To ensure all devices are biased in saturation, the following operating point constraints were added to the design specification file.

$$V_{DS1} \geq V_{EB1} \tag{240}$$

$$V_{SD2} \geq V_{EB2} \tag{241}$$

$$V_{DS3} \geq V_{EB3} \tag{242}$$

$$V_{SD4} \geq V_{EB4} \tag{243}$$

$$V_{DS5} \geq V_{EB5} \tag{244}$$

Device Widths

The device widths are determined by solving the large signal equations of (54) and (55) for W.

$$W_1 = \frac{I_{DS1} L_{eff1}}{\beta_n V_{EB1}^2 \left(1 + \lambda_n V_{DS1}\right)} \tag{245}$$

$$W_2 = \frac{I_{SD2} L_{eff2}}{\beta_p V_{EB2}^2 \left(1 + \lambda_p V_{SD2}\right)} \tag{246}$$

$$W_3 = \frac{I_{DS3} L_{eff3}}{\beta_n V_{EB3}^2 \left(1 + \lambda_n V_{DS3}\right)} \tag{247}$$

$$W_4 = \frac{I_{SD4} L_{eff4}}{\beta_p V_{EB4}^2 \left(1 + \lambda_p V_{SD4}\right)} \tag{248}$$

$$W_5 = \frac{I_{DS5} L_{eff5}}{\beta_n V_{EB5}^2 \left(1 + \lambda_n V_{DS5}\right)} \tag{249}$$

Small-Signal Model

Fig. 65 shows the small-signal model of the fully-differential telescopic cascode amplifier shown in Fig. 64. To simplify the model, the parallel capacitances are lumped together as described in equations (250) through (253).

$$c_0 = c_L + c_{gd4} + c_{db5} + c_{gd5} \tag{250}$$

$$c_1 = c_{db1} + c_{sb5} + c_{gs5} \tag{251}$$

$$c_2 = c_{dw2} + c_{gd2} + c_{wb4} + c_{gs4} \tag{252}$$

$$c_3 = c_{db3} + c_{gd3} + 2 \cdot c_{sb1} \tag{253}$$

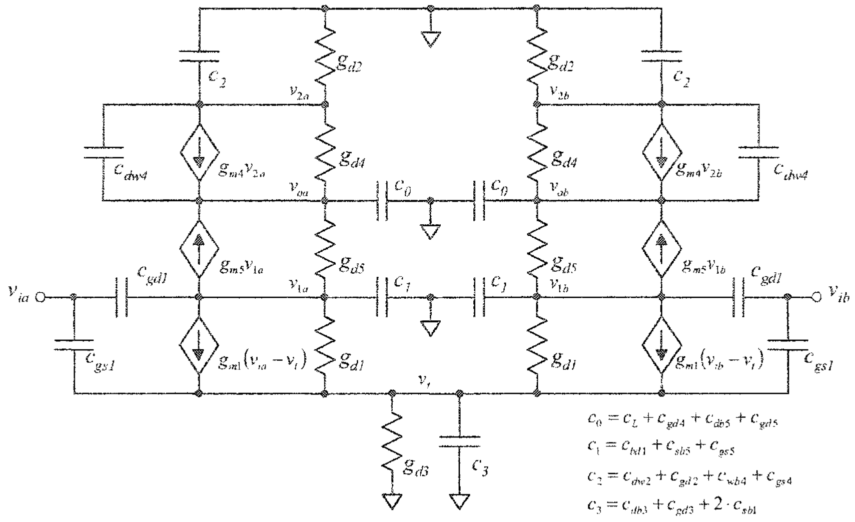


Fig. 65 Small-signal model of the fully-differential telescopic cascode amplifier

For differential excitation, the tail node acts as a virtual ground and the small-signal model of Fig. 65 reduces to the one shown in Fig. 66.

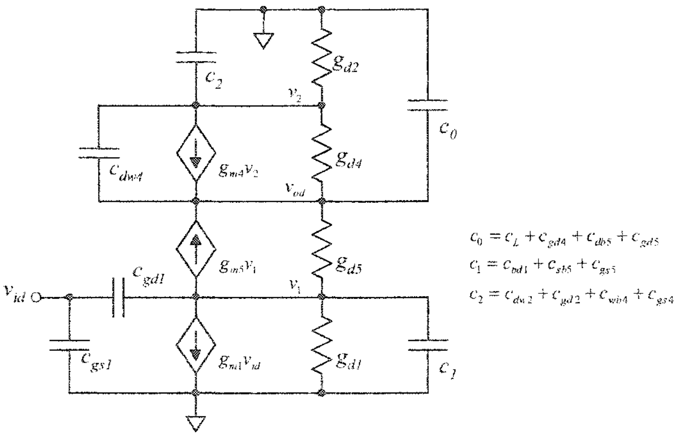


Fig. 66 Simplification of small-signal model assuming differential excitation

The circuit of Fig. 66 is small enough that its analytical response can be determined without the assistance of a computer. However, the task is arduous and susceptible to errors. Upon completion, the resultant expressions are too complicated to be tractable. Because the expressions are so complicated, factorizations are unlikely to be obtained by hand without making simplifying assumptions (which sacrifice accuracy). For these reasons, hand calculations were avoided, and the exact analytical response was computed using the computer arithmetic software package Maple[2]. Table 49 contains the Maple script used to determine the differential transfer function. When the script is executed, Maple computes the transfer function and generates the optimized Fortran code required to implement it. Table 50 contains a listing of the resultant Fortran code generated by Maple. These expressions are easily adapted for use with Design Space Explorer using a text editor.

Table 49 Maple script used to generate the code to compute the differential response

```
#
# Maple script to determine the differential response of a telescopic
# cascode amplifier.
#
eq1 := gm1*vid + v1*(s*c1+gd1) + (v1-vid)*s*cgd1 + gm5*v1 + (v1-vod)*gd5;
eq2 := gm4*v2 + (v2-vod)*(gd4+s*cdw4) + v2*(s*c2+gd2);
eq3 := (vod-v2)*(s*cdw4+gd4) - gm4*v2 - gm5*v1 + (vod-v1)*gd5 + vod*s*c0;

v1 := solve(eq1,v1);
v2 := solve(eq2,v2);

hd:=solve(eq3,vod)/vid;
d:=collect(denom(hd),s,factor);
n:=collect(numer(hd),s,factor);

a0t := subs(s=0,n);
a1t := coeff(n,s^1);
a2t := coeff(n,s^2);
b0t := subs(s=0,d);
b1t := coeff(d,s^1);
b2t := coeff(d,s^2);
b3t := coeff(d,s^3);

# coefficients of the open-loop transfer function
fortran([a0=a0t,a1=a1t,a2=a2t,b0=b0t,b1=b1t,b2=b2t,b3=b3t],optimized);
```

[2] Maple is a trademark of Waterloo Maple Inc.

Table 50 Fortran code that computes the coefficients of the differential transfer function

```
 t1 = gd5+gm5
 a0 = -gm1*t1*(gm4+gd4+gd2)
 a1 = -(gm1*cdw4+gm1*c2-cgd1*gm4-gd4*cgd1-cgd1*gd2)*t1
 a2 = cgd1*t1*(cdw4+c2)
 t14 = gd5*gd1
 t16 = gd2*gd4
 t21 = gd4*gd5
 b0 = t14*gd2+t16*gd1+t16*gd5+t14*gm4+t16*gm5+t21*gd1
 t23 = c2*gd4
 t25 = c0*gm5
 t27 = gd5*c1
 t29 = gd2*cdw4
 t31 = gd5*cgd1
 t41 = c0*gd4
 t43 = t23*gd5+t25*gm4+t27*gm4+t29*gd1+t31*gm4+t29*gd5+t31*gd2+t16*
#cgd1+c0*gm4*gd5+c0*gd2*gd5+t25*gd2+t41*gd1
 t45 = gd5*cdw4
 t49 = c0*gd1
 t59 = t23*gd1+t45*gd1+t27*gd2+t16*c1+t49*gd2+t14*c2+t41*gd5+t29*gm
#5+t41*gm5+t49*gm4+t21*cgd1+t21*c1+t23*gm5
 b1 = t43+t59
 t60 = c0*c1
 t65 = c0*cdw4
 t67 = c2*cdw4
 t72 = c0*cgd1
 t75 = t60*gd2+t45*cgd1+t29*cgd1+t23*c1+t65*gd1+t67*gd5+t49*c2+t29*
#c1+t41*cgd1+t72*gd2+t67*gm5
 t85 = c0*c2
 t89 = t65*gd5+t72*gm4+t23*cgd1+t31*c2+t60*gm4+t25*c2+t27*c2+t45*c1
#+t67*gd1+t85*gd5+t41*c1+t65*gm5
 b2 = t75+t89
 b3 = (t65+t85+t67)*(cgd1+c1)
```

The resultant open-loop differential transfer function is of the form:

$$H_{ol}(s)=\frac{a_2 s^2+a_1 s+a_0}{b_3 s^3+b_2 s^2+b_1 s+b_0} \tag{254}$$

where the numerator and denominator coefficients are computed using the code in Table 50. As discussed in 0, the common-mode transfer function is usually not of interest when fully-differential signaling is employed. The small-signal transconductances were computed using (61) and (62) while the output conductances were determined using (63) and (64).

To compute the frequency response, the capacitive parasitics must be determined. In order to accurately estimate the parasitics, knowledge of the specific layout is required. Since the transistor layouts are typically not available at the design space exploration phase, specific layout styles were assumed for the transistors. An effort was made to choose appropriate layout styles. However, if they are not suitable for a specific application the diffusion area and perimeter calculations will need to be modified.

M1A and M1B are assumed to be laid-out using the scheme of 0. M2A and M2B using the technique of 0. M3 is laid-out using the technique described in 0 and M4A, M4B, M5A, and M5B are laid out using the technique of 0. Tables 51 and 52 show the calculations used to compute the diffusion perimeters and areas. Table 53 shows the calculations used to determine the number of fingers in the transistor layouts that most closely achieve the desired aspect ratios.

Table 51 Diffusion perimeter calculations for the telescopic cascode amplifier

Name	Diffusion	Perimeter
P_{D1}	D1A, D1B	$W_1 + NF_1 \cdot \beta$
P_{S1}	S1	$2 \cdot \left(\frac{NF_1 + 1}{NF_1} \right) P_{D1}$
P_{D2}	D2A, D2B	$W_2 + NF_2 \cdot \beta$
P_{D3}	D3	$W_3 + NF_3 \cdot \beta$
P_{D4}	D4A, D4B	$W_4 + NF_4 \cdot \beta$
P_{S4}	S4A, S4B	$2 \cdot (W_4 + NF_4 \cdot \beta')$
P_{D5}	D5A, D5B	$W_5 + NF_5 \cdot \beta$
P_{S5}	S5A, S5B	$2 \cdot (W_5 + NF_5 \cdot \beta')$

Table 52 Diffusion area calculations for the telescopic cascode amplifier

Name	Diffusion	Area
A_{D1}	D1A, D1B	$\frac{W_1 \cdot \beta}{2}$
A_{S1}	S1	$2 \cdot \left(\frac{NF_1 + 1}{NF_1} \right) A_{D1}$
A_{D2}	D2A, D2B	$\frac{W_2 \cdot \beta}{2}$
A_{D3}	D3	$\frac{W_3 \cdot \beta}{2}$
A_{D4}	D4A, D4B	$\frac{W_4 \cdot \beta}{2}$
A_{S4}	S4A, S4B	$W_4 \cdot \beta'$
A_{D5}	D5A, D5B	$\frac{W_5 \cdot \beta}{2}$
A_{S5}	S5A, S5B	$W_5 \cdot \beta'$

Table 53 Calculations to determine the number of fingers/transistor

Name	Calculation
β	$szCont + 2 \cdot spcContGate$
α_k	$\dfrac{\beta}{4(L_k + \beta)}, \qquad k \in \{1,2,3\}$
ξ_i	$\dfrac{\kappa_i}{\mathrm{K} + \kappa_i}, \qquad i \in \{4,5\}$
K_i	$2 \cdot (ovActCont + \beta + L_i) + szCont, \qquad i \in \{4,5\}$
κ_4	$2 \cdot spcSDActWell + spcWellDiffPot$
κ_5	$spcAct$
NF_1	$\max\left(4 \cdot \mathrm{rint}\left(\alpha_1 \cdot \left(\sqrt{1 + \dfrac{2 \cdot W_1 \cdot DAR_1}{\alpha_1 \cdot \beta}} - 1\right)\Big/4\right), 4\right)$
NF_2	$\max\left(4 \cdot \mathrm{rint}\left(\alpha_2 \cdot \left(\sqrt{1 + \dfrac{2 \cdot W_2 \cdot DAR_2}{\alpha_2 \cdot \beta}} - 1\right)\Big/4\right), 4\right)$
NF_3	$\max\left(2 \cdot \mathrm{rint}\left(2 \cdot \alpha_3 \cdot \left(\sqrt{1 + \dfrac{W_3 \cdot DAR_3}{\alpha_3 \cdot \beta}} - 1\right)\Big/2\right), 2\right)$
NF_4	$\max\left(4 \cdot \mathrm{rint}\left(\xi_4 \left\{1 + \sqrt{1 + \dfrac{2 \cdot W_4 \cdot DAR_4}{\kappa_4 \cdot \xi_4}}\right\}\Big/4\right), 4\right)$
NF_5	$\max\left(4 \cdot \mathrm{rint}\left(\xi_5 \left\{1 + \sqrt{1 + \dfrac{2 \cdot W_5 \cdot DAR_5}{\kappa_5 \cdot \xi_5}}\right\}\Big/4\right), 4\right)$

The procedure used to estimate the parasitic capacitances of a FET operating in saturation is described in section 0. Table 54 details the equations used to compute the sizes of the relevant parasitic elements. The parasitic elements that have fixed DC potentials across them have been omitted. They are shorted in the small-signal model and therefore can be neglected.

Table 54 Parasitic capacitance calculations for telescopic cascode amplifier

Name	Capacitance Calculation
C_{gdk}	$C_{oxn}W_kL_D, \qquad k \in \{1,3,5\}$
C_{gdk}	$C_{oxp}W_kL_D, \qquad k \in \{2,4\}$
C_{gsk}	$C_{oxn}W_kL_D + \frac{2}{3}C_{oxn}W_kL_k, \qquad k \in \{1,5\}$
C_{gs4}	$C_{oxp}W_4L_D + \frac{2}{3}C_{oxp}W_4L_4$
C_{dw2}	$\frac{C_{JP}A_{D2}}{(1+V_{SD2}/\phi_{BP})^{MJP}} + \frac{C_{JSWP}P_{D2}}{(1+V_{SD2}/\phi_{BP})^{MJSWP}}$
C_{dw4}	$\frac{C_{JP}A_{D4}}{(1+V_{SD4}/\phi_{BP})^{MJP}} + \frac{C_{JSWP}P_{D4}}{(1+V_{SD4}/\phi_{BP})^{MJSWP}}$
C_{wb4}	$\frac{C_{JP}A_{well4}}{(1+(V_2-V_{SS})/\phi_{BP})^{MJP}} + \frac{C_{JSWP}P_{well4}}{(1+(V_2-V_{SS})/\phi_{BP})^{MJSWP}}$
C_{db1}	$\frac{C_{JN}A_{D1}}{(1+(V_1-V_{SS})/\phi_{BN})^{MJN}} + \frac{C_{JSWN}P_{D1}}{(1+(V_1-V_{SS})/\phi_{BN})^{MJSWN}}$
C_{db3}	$\frac{C_{JN}A_{D3}}{(1+V_{DS3}/\phi_{BN})^{MJN}} + \frac{C_{JSWN}P_{D3}}{(1+V_{DS3}/\phi_{BN})^{MJSWN}}$
C_{db5}	$\frac{C_{JN}A_{D5}}{(1+(V_O-V_{SS})/\phi_{BN})^{MJN}} + \frac{C_{JSWN}P_{D5}}{(1+(V_O-V_{SS})/\phi_{BN})^{MJSWN}}$
C_{sb1}	$\frac{C_{JN}A_{S1}}{(1+V_{DS3}/\phi_{BN})^{MJN}} + \frac{C_{JSWN}P_{S1}}{(1+V_{DS3}/\phi_{BN})^{MJSWN}}$
C_{sb5}	$\frac{C_{JN}A_{S5}}{(1+(V_1-V_{SS})/\phi_{BN})^{MJN}} + \frac{C_{JSWN}P_{S5}}{(1+(V_1-V_{SS})/\phi_{BN})^{MJSWN}}$

Open-Loop Response

The open-loop differential transfer function of (254) can be factorized into the form of (255) using the *rootpoly()* function.

$$H_{ol} = \frac{A_{ol} \cdot (1 - s/Z_1) \cdot (1 - s/Z_2)}{(1 - s/P_{ol1}) \cdot (1 - s/P_{ol2}) \cdot (1 - s/P_{ol3})} \tag{255}$$

The locations of the poles and zeros are determined via (256) and (257) respectively.

$$\{P_{ol1}, P_{ol2}, P_{ol3}\} = rootpoly(b_0, b_1, b_2, b_3) \tag{256}$$

$$\{Z_1, Z_2\} = rootpoly(a_0, a_1, a_2) \tag{257}$$

The DC gain is computed by substituting s=0 in (254).

$$A_{ol} = \frac{a_0}{b_0} \tag{258}$$

Closed-Loop Response

If the telescopic cascode amplifier of Fig. 64 is used as the amplifier in a feedback configuration, then the closed-loop transfer function can be found by substituting the numerator and denominator from (254) into (143).

$$H_{cl}(s) = \frac{a_2 s^2 + a_1 s + a_0}{b_3 s^3 + b_2 s^2 + b_1 s + b_0 + \beta (a_2 s^2 + a_1 s + a_0)} \tag{259}$$

Which can also be expressed as:

$$H_{cl}(s) = \frac{a_2 s^2 + a_1 s + a_0}{d_3 s^3 + d_2 s^2 + d_1 s + d_0} \tag{260}$$

where

$$d_3 = b_3 \tag{261}$$

$$d_2 = b_2 + \beta \cdot a_2 \tag{262}$$

$$d_1 = b_1 + \beta \cdot a_1 \tag{263}$$

$$d_0 = b_0 + \beta \cdot a_0 \tag{264}$$

The poles migrate as the feedback factor varies. For a given feedback factor, the closed-loop pole locations are given by:

$$\{P_{cl1}, P_{cl2}, P_{cl3}\} = rootpoly(d_0, d_1, d_2, d_3) \tag{265}$$

The closed-loop DC gain is computed by substituting *s*=0 in (259).

$$A_{cl} = \frac{a_0}{d_0} \tag{266}$$

To ensure that only stable closed-loop systems are considered, constraints were added to the design specification file to ensure that all the closed-loop poles lie in the left half-plane. Using the variables defined in (265) and (266), the transfer function of (259) can be expressed as:

$$H_{cl}(s) = \frac{A_{cl} \cdot (1 - s/Z_1) \cdot (1 - s/Z_2)}{(1 - s/P_{cl1}) \cdot (1 - s/P_{cl2})(1 - s/P_{cl3})} \tag{267}$$

Slew-Rate

Due to their architectural similarity, the slew-rate analysis of the standard fully differential amplifier given in section 0 applies to the telescopic cascode amplifier as well. The slew-rate for the telescopic cascode can be written as:

$$SR = \frac{I_{DS3}}{c_0} \tag{268}$$

where I_{DS3} is the tail current and c_0 is the output node capacitance given in (224) and (250) respectively.

Linear Step Response

The closed-loop system's transient step response computed using the inverse Laplace Transform is given in (269).

$$v_{od}(t) = A_{cl} \cdot \xi \cdot \left\{1 + k_1 \cdot e^{P_{cl1} \cdot t} + k_2 \cdot e^{P_{cl2} \cdot t} + k_3 \cdot e^{P_{cl3} \cdot t}\right\} \tag{269}$$

The variable ξ represents the magnitude of the input step and the coefficients of the exponentials are given by (270)-(272).

$$k_1 = -\frac{(1 - P_{cl1}/Z_1)(1 - P_{cl1}/Z_2)}{(1 - P_{cl1}/P_{cl2})(1 - P_{cl1}/P_{cl3})} \tag{270}$$

$$k_2 = -\frac{(1 - P_{cl2}/Z_1)(1 - P_{cl2}/Z_2)}{(1 - P_{cl2}/P_{cl1})(1 - P_{cl2}/P_{cl3})} \tag{271}$$

$$k_3 = -\frac{(1 - P_{cl3}/Z_1)(1 - P_{cl3}/Z_2)}{(1 - P_{cl3}/P_{cl1})(1 - P_{cl3}/P_{cl2})} \tag{272}$$

Due to nonlinearity, the real transient response and (269) will not exactly agree. For example, for large input steps, the amplifier may initially slew toward the final value. Furthermore, as the node voltages and branch currents deviate from their quiescent values with signal swing, the small-signal parameters of the devices vary. Despite the fact that the response of (269) is inexact, designers still find it useful to look at these expressions nonetheless.

The Two-Stage Amplifier

This section describes the design specification file for the fully-differential two-stage amplifier shown in Fig. 67. The first stage consists of transistors M1A, M1B, M2A, M2B, and M3 while the second stage is made up by M4A, M4B, M5A and M5B. The compensation network is comprised of capacitors C_{CA} and C_{CB} and the resistors R_{CA} and R_{CB}.

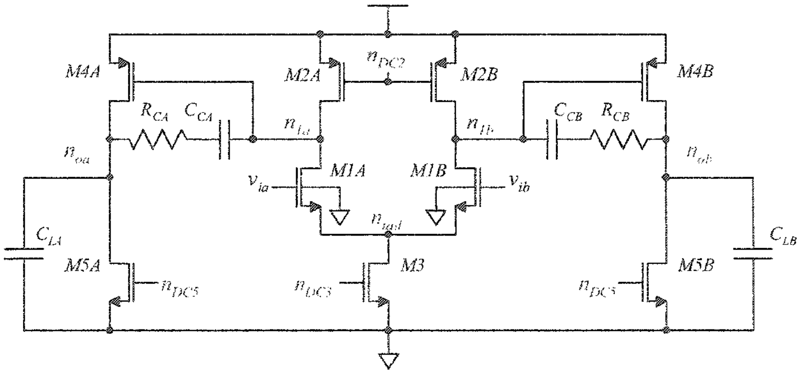


Fig. 67 Schematic diagram of a two-stage amplifier

Degrees of Freedom

The degrees of freedom used for the two-stage amplifier are listed in Table 55.

Table 55 Degrees of freedom used to characterize the two-stage amplifier

V_{EB1}, V_{EB2}, V_{EB3}, V_{EB4}, V_{EB5}	Excess biases of devices M1-M5 (V)
L_1, L_2, L_3, L_4, L_5	Lengths of devices M1-M5 (m)
P	Total power consumption (W)
V_{DD}, V_{SS}	Supply voltages (V)
V_{ICM}	Common-mode input voltage (V)
C_L	Capacitive load (F)
DAR1, DAR2, DAR3, DAR4, DAR5	Desired aspect ratios of devices M1-M5
BETA	Feedback factor, for closed-loop configuration
h	Settling accuracy parameter

Common-Mode Output Voltage

The upper limit on the common-mode output voltage is imposed by M4A and M4B. The maximum common-mode output voltage for which they operate in saturation is given by:

$$V_{OCM(\max)} = V_{DD} - V_{EB4} \tag{273}$$

where V_{1Q} is the quiescent voltage on nodes n_{1a} and n_{1b}. Likewise, the lower limit is imposed by M5A and M5B. To operate in saturation, the common-mode output voltage must not drop below a certain level.

$$V_{OCM(\min)} = V_{EB5} \tag{274}$$

To ensure only viable designs are considered, the following constraint was added to the design specification file.

$$V_{OCM(\min)} \leq V_{OCM} \leq V_{OCM(\max)} \tag{275}$$

The choice of common-mode output voltage affects the magnitude of the maximum undistorted differential output signal that the amplifier can produce. To ensure a large differential output signal swing capability, the common-mode output voltage is normally chosen to be near the middle of the acceptable range. However, since the required common-mode output voltage is commonly specified by the application, it was included as a degree of freedom in the design specification file.

Maximum Differential Output Swing

Normally, the transistors associated with the second stage are the ones that limit the maximum differential output signal swing. For a perfectly matched circuit with a differential input of zero volts, the voltages at n_{oa} and n_{ob} are equal. As the magnitude of the differential input voltage increases, v(n_{oa}) or v(n_{ob}) increases while the other decreases. At some point, the magnitudes of the deviations become so large that one or more of the transistors leaves saturation. The relationships are illustrated in Fig. 68.

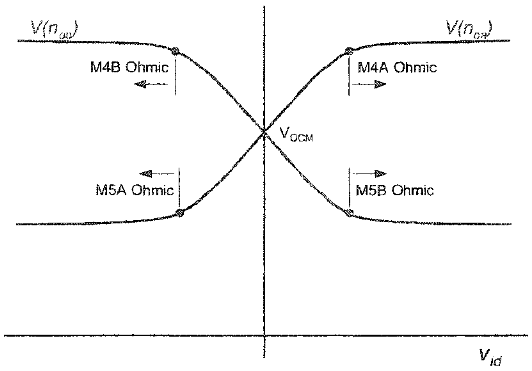


Fig. 68 Large-signal output node voltages as a function of the differential input voltage

Requiring that M5A and M5B operate in saturation determines the minimum voltages at the output nodes.

$$V_{oa(\min)} = V_{EB5} \tag{276}$$

The points where M4A and M4B leave saturation correspond to the maximum nodal output voltage. Using the large-signal expressions and solving for the exact points where M4A and M4B leave saturation yields complicated analytical expressions. Although coding the required expressions is possible in Design Space Explorer, knowing the exact value of the maximum output voltage is not necessary. Errors as large as a few percent of the total output voltage swing are acceptable. For this a reason, a simple approximation of the maximum output voltage is employed. Fig. 69 illustrates the technique used to estimate the point where M4A leaves saturation.

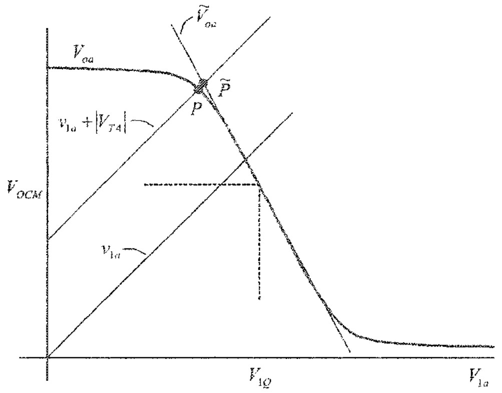


Fig. 69 Approximating the point where M4A leaves saturation

Fig. 69 shows the transfer characteristic of the second-stage (M4A and M4B). The maximum output voltage, labeled *P*, is the point where M4A leaves saturation. Mathematically, *P* is the point where (277) is satisfied.

$$V_{oa} = V_{1a} + |V_{T4}| \tag{277}$$

Thus P lies at the intersection of the line $V_{1a} + |V_{T4}|$ and the transfer characteristic. Since the expression for V_{oa} is cumbersome, finding P can be troublesome. Since the nearby point, $\widetilde{P}$, is a good approximation of P and it is found with much less difficulty, $\widetilde{P}$ is used instead. $\widetilde{P}$ is found by approximating V_{oa} in (277) with the linear approximation of (278).

$$\widetilde{V}_{oa} = V_{OCM} + A_2 \cdot (V_{1a} - V_{1Q}) \tag{278}$$

The variables V_{OCM} and V_{1Q} represent the quiescent voltages at nodes n_{oa} and n_{1a} respectively and A_2 is the small-signal gain evaluated at the quiescent point.

$$A_2 = \frac{-g_{m4}}{g_{d4} + g_{d5}} \tag{279}$$

Solving for $\widetilde{P}$ results yields an approximate expression for the maximum output voltage.

$$V_{oa(\max)} \approx \frac{V_{OCM} - A_2(V_{DD} - V_{EB4})}{1 - A_2} \tag{280}$$

The maximum positive voltage excursion on either of the output nodes from the common-mode voltage is given by:

$$\Delta V_{o+} = V_{oa(\max)} - V_{OCM} \tag{281}$$

Likewise the magnitude of the largest negative voltage excursion on either of the output nodes from the common-mode value is:

$$\Delta V_{o-} = V_{OCM} - V_{oa(\min)} \tag{282}$$

This corresponds to the maximum single-ended swing at n_{oa} or n_{ob}. The maximum peak-peak differential swing is twice as large.

$$V_{od(\max)} = 4 \cdot \min\left(\Delta V_{o+}, \Delta V_{o-}\right) \tag{283}$$

Operating Point

The total DC current consumed by the amplifier is given by:

$$I_{total} = I_{DS3} + 2I_{SD4} = \frac{P}{V_{DD} - V_{SS}} \tag{284}$$

The ratio of current consumed by the output stage to that of the input stage is given by θ.

$$\theta = \frac{I_{SD4A} + I_{SD4B}}{I_{DS3}} = \frac{2I_{SD4}}{I_{DS3}} \tag{285}$$

The remaining device currents can be expressed in terms of I_{total} and θ as summarized in Table 56.

Table 56 Operating point currents

Current	*Value*	*Description*
I_{DS1}, I_{SD2}	$\frac{I_{DS3}}{2}$	M1 drain to source and M2 source to drain currents
I_{DS3}	$\frac{I_{total}}{(1+\theta)}$	M3 drain to source current (Tail Current)
I_{SD4}	$\frac{\theta \cdot I_{total}}{2(1+\theta)}$	M4 source to drain current
I_{DS5}	I_{SD4}	M5 drain to source current

Assuming the common-mode input and output voltages are known, all the node voltages can be determined. The tail voltage can be expressed as:

$$V_{TAIL} = V_{ICM} - V_{GS1} = V_{ICM} - \left(V_{EB1} + V_{T1}\right) \tag{286}$$

where V_{ICM} is the common-mode input voltage, V_{EB1} is M1's excess bias voltage, and V_{T1} is M1's threshold. Due to the body-effect, the threshold voltage has a slight dependence on V_{TAIL}. Since the threshold voltage is not constant, (286) is not an explicit expression for V_{TAIL}. Finding the exact value of the threshold requires the solution of a nonlinear equation. The amount of variation in the tail voltage, however, induced due to the body-effect is very small. Neglecting the dependency does not significantly impact the accuracy of the model. Therefore, the M1's body-effect has been ignored.

The voltages on nodes n_{1a} and n_{1b} can be expressed as:

$$V_1 = V_{DD} - V_{SG4} = V_{DD} - \left(V_{EB4} + \left|V_{T4}\right|\right) \tag{287}$$

The voltage on the output node is assumed to be equal to the common-mode output voltage V_{OCM}. The circuit's nodal voltages are summarized in Table 58.

Table 58 Nodal voltages at the operating point

Quantity	*Node(s)*	*Value*	*Description*
V_o	n_{oa}, n_{ob}	V_{OCM}	Output
V_i	n_{ia}, n_{ib}	V_{ICM}	Input
V_{tail}	n_{tail}	$V_{ICM} - V_{GS1}$	Tail
V_1	n_{1a}, n_{1b}	$V_{DD} - V_{SG4}$	intermediate
V_{DC2}	n_{DC2}	$V_{DD} - V_{SG2}$	Bias
V_{DC3}	n_{DC3}	$V_{SS} + V_{GS3}$	Bias
V_{DC5}	n_{DC5}	$V_{SS} + V_{GS5}$	Bias

Table 59 lists the device specific operating point voltages. Using these quantities in hierarchical calculations can reduce the computational complexity of the model (see 0) and improve the readability of the design specification file.

Table 59 Device specific operating point voltages

Quantity	Value	Description
V_{GSk}	$V_{EBk} + V_{Tk}$	Gate-source voltage, $k \in \{1,3,5\}$
V_{SGk}	$V_{EBk} - V_{Tk}$	Source-gate voltage, $k \in \{2,4\}$
V_{DS1}	$V_1 - V_{tail}$	M1 drain-source voltage
V_{SD2}	$V_{DD} - V_1$	M2 source-drain voltage
V_{DS3}	$V_{tail} - V_{SS}$	M3 drain-source voltage
V_{SD4}	$V_{DD} - V_O$	M4 source-drain voltage
V_{DS5}	$V_O - V_{SS}$	M5 drain-source voltage

Constraints were added to the design specification file to eliminate solutions for which one or more device is not operating in saturation.

Device Widths

The device widths are determined by solving the large signal equations of (54) and (55) for W. The expressions used to compute the device widths are listed in Table 60.

Table 60 Device width calculations

Quantity	*Value*	*Description*
W_k	$\dfrac{I_{DSk}L_{effk}}{\beta_n V_{EBk}^2\left(1+\lambda_n V_{DSk}\right)}$	Width of device k, $k \in \{1{,}3{,}5\}$
W_k	$\dfrac{I_{SDk}L_{effk}}{\beta_p V_{EBk}^2\left(1+\lambda_p V_{SDk}\right)}$	Width of device k, $k \in \{2{,}4\}$

Small-Signal Model

For differential excitation, the tail node acts as a virtual ground. Under these conditions, the small-signal model of the overall circuit is equivalent to the small-signal model of the half-circuit. Fig. 70 contains a schematic diagram of the small-signal model. To simplify the model, parallel capacitances and conductances are lumped together as described in equations (288) through (292).

$$c_1 = c_{db1} + c_{dw2} + c_{gd2} + c_{gs4} \tag{288}$$

$$c_O = c_L + c_{dw4} + c_{db5} + c_{gd5} \tag{289}$$

$$g_{dd} = g_{d1} + g_{d2} \tag{290}$$

$$g_{do} = g_{d4} + g_{d5} \tag{291}$$

$$g_c = 1/R_C \tag{292}$$

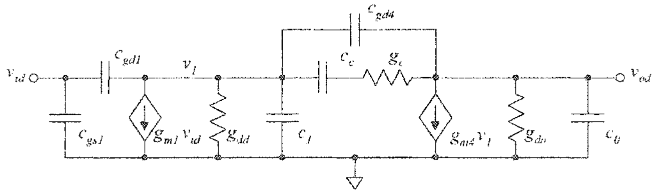


Fig. 70 Small signal model of a two-stage amplifier

As in the case of the telescopic cascode amplifier, the small-signal model of Fig. 70 is small enough that its analytical response can be determined without the assistance of a computer. However, the task is arduous and susceptible to errors. Upon completion, the resultant expressions are too complicated to be tractable. Because the expressions are so complicated, factorizations are unlikely to be obtained by hand without making simplifying assumptions (which sacrifice accuracy). For these reasons, hand calculations were avoided, and the exact analytical response was computed using the computer arithmetic software package Maple[3]. Table 61 contains the Maple script used to determine the differential transfer function. When the script is executed, Maple computes the transfer function and generates the optimized Fortran code required to implement it. Table 62 contains a listing of the resultant Fortran code generated by Maple. These expressions are easily adapted for use with Design Space Explorer using a text editor.

Table 61 Maple script used to generate the code to compute the differential response

[3] Maple is a trademark of Waterloo Maple Inc.

```
# Maple script to analyze the small-signal response of a two-stage
amplifier
# Includes a resistor in series with the miller compensation capacitor
# to flip the rhp zero into the lhp
# admittance of compensation network
yc:=s*cgd4+1/(1/gc+1/s/cc):;
# sum currents at node n1
s*cgd1*(v1-vid)+gm1*vid+(gdd+s*c1)*v1+(v1-vod)*yc:;
v1:=solve(%,v1):;
# sum currents at output
(vod-v1)*yc+gm4*v1+(gdo+s*c0)*vod:;
solve(%,vod):;
vod:=collect(%,s):;
Hd:=vod/vid:;
n:=collect(numer(Hd),s);
d:=collect(denom(Hd),s,factor);
# denominator coefficients
b0t := subs(s=0,d);
b1t := coeff(d,s^1);
b2t := coeff(d,s^2);
b3t := coeff(d,s^3);
# numerator coefficients
factor(n);
nterm1:=op(1,%);
nterm2:=op(2,%%);
n11:=coeff(nterm1,s);
n10:=subs(s=0,nterm1);
n22:=coeff(nterm2,s^2);
n21:=coeff(nterm2,s);
n20:=subs(s=0,nterm2);
# coefficients of the open-loop transfer function
fortran([a20=n10,a21=n11,a10=n20,a11=n21,a12=n22,b0=b0t,b1=b1t,b2=b2t,b3=b
3t],optimized);
```

Table 62 Fortran code that computes the coefficients of the differential transfer function

```
 a20 = -gm1
 a21 = cgd1
 a10 = -gm4*gc
 t4 = gm4*cc
 a11 = cgd4*gc+gc*cc-t4
 a12 = cgd4*cc
 b0 = gdo*gdd*gc
 t6 = cc*gdo
 t9 = gdd*cc
 b1 = t6*gdd+t4*gc+t9*gc+t6*gc+gdo*cgd4*gc+gdo*cgd1*gc+gdo*c1*gc+gd
#d*cgd4*gc+c0*gdd*gc+gm4*cgd4*gc
 t31 = cc*c0
 t36 = c0*cgd4
 t38 = c1*cgd4
 t40 = c0*cgd1
 t42 = c0*c1
 t44 = cgd1*cgd4
 b2 = t4*cgd4+t6*cgd4+c1*cc*gc+cgd1*cc*gc+t6*cgd1+t31*gc+t9*cgd4+t3
#1*gdd+t6*c1+t36*gc+t38*gc+t40*gc+t42*gc+t44*gc
 b3 = (t44+t40+t42+t36+t38)*cc
```

The resultant open-loop differential transfer function is of the form:

$$H_{ol}(s) = \frac{(a_{21}s + a_{20})(a_{12}s^2 + a_{11}s + a_{10})}{(b_3 s^3 + b_2 s^2 + b_1 s + b_0)} \tag{293}$$

where the numerator and denominator coefficients are computed using the code in Table 62. As discussed in 0, the common-mode transfer function is usually not of interest when fully-differential signaling is employed. The small-signal transconductances were computed using (61) and (62) while the output conductances were determined using (63) and (64). Fully expanding the product in the numerator, the transfer function can be written as:

$$H_{ol}(s) = \frac{(n_3 s^3 + n_2 s^2 + n_1 s + n_0)}{(b_3 s^3 + b_2 s^2 + b_1 s + b_0)} \tag{294}$$

where the numerator coefficients are given by:

$$n_3 = a_{21}a_{12} \tag{295}$$

$$n_2 = \left(a_{21}a_{11} + a_{20}a_{12}\right) \tag{296}$$

$$n_1 = \left(a_{21}a_{10} + a_{20}a_{11}\right) \tag{297}$$

$$n_0 = a_{20}a_{10} \tag{298}$$

To compute the frequency response, the capacitive parasitics must be determined. In order to accurately estimate the parasitics, knowledge of the specific layout is required. Since the transistor layouts are typically not available at the design space exploration phase, specific layout styles were assumed for the transistors. An effort was made to choose appropriate layout styles. However, if they are not suitable for a specific application the diffusion area and perimeter calculations will need to be modified.

M1A and M1B are assumed to be laid-out using the scheme of 0. M2A, M2B, M5A, and M5B using the technique of 0. M3, M4A, and M4B are laid-out using the technique described in 0. Tables 63 and 64 show the calculations used to compute the diffusion perimeters and areas. Table 65 shows the calculations used to determine the number of fingers in the transistor layouts that most closely achieve the desired aspect ratios.

Table 63 Diffusion perimeter calculations for the two-stage amplifier

Name	Diffusion	Perimeter
P_{D1}	D1A, D1B	$W_1 + NF_1 \cdot \beta$
P_{S1}	S1	$\left(\frac{NF_1 + 1}{NF_1}\right)P_{D1}$
P_{D2}	D2A, D2B	$W_2 + NF_2 \cdot \beta$
P_{D3}	D3	$W_3 + NF_3 \cdot \beta$
P_{D4}	D4A, D4B	$W_4 + NF_4 \cdot \beta$
P_{D5}	D5A, D5B	$W_5 + NF_5 \cdot \beta$

Table 64 Diffusion area calculations for the two-stage amplifier

Name	Area
A_{Dk}	$\frac{W_k \cdot \beta}{2}, \qquad k \in \{1,2,3,4,5\}$
A_{S1}	$\left(\frac{NF_1 + 1}{NF_1}\right) A_{D1}$

Table 65 Calculations to determine the number of fingers/transistor

Name	Calculation
β	$szCont + 2 \cdot spcContGate$
α_k	$\frac{\beta}{4(L_k + \beta)}, \qquad k \in \{1,2,3,4,5\}$
NF_1	$\max\left(4 \cdot \mathrm{rint}\left(\alpha_1 \cdot \left(\sqrt{1 + \frac{2 \cdot W_1 \cdot DAR_1}{\alpha_1 \cdot \beta}} - 1\right) \Big/ 4\right), 4\right)$
NF_2	$\max\left(4 \cdot \mathrm{rint}\left(\alpha_2 \cdot \left(\sqrt{1 + \frac{2 \cdot W_2 \cdot DAR_2}{\alpha_2 \cdot \beta}} - 1\right) \Big/ 4\right), 4\right)$
NF_3	$\max\left(2 \cdot \mathrm{rint}\left(2 \cdot \alpha_3 \cdot \left(\sqrt{1 + \frac{W_3 \cdot DAR_3}{\alpha_3 \cdot \beta}} - 1\right) \Big/ 2\right), 2\right)$
NF_4	$\max\left(2 \cdot \mathrm{rint}\left(2 \cdot \alpha_4 \cdot \left(\sqrt{1 + \frac{W_4 \cdot DAR_4}{\alpha_4 \cdot \beta}} - 1\right) \Big/ 2\right), 2\right)$
NF_5	$\max\left(4 \cdot \mathrm{rint}\left(\alpha_5 \cdot \left(\sqrt{1 + \frac{2 \cdot W_5 \cdot DAR_5}{\alpha_5 \cdot \beta}} - 1\right) \Big/ 4\right), 4\right)$

The procedure used to estimate the parasitic capacitances of a FET operating in saturation is described in section 0. Table 66 details the equations used to compute the sizes of the relevant parasitic elements. The parasitic elements that have fixed DC potentials across them have been omitted. They are shorted in the small-signal model and therefore can be neglected.

Table 66 Parasitic capacitance calculations for the two-stage amplifier

Name	Capacitance Calculation
C_{gdk}	$C_{oxn}W_kL_D, \qquad k \in \{1,3,5\}$
C_{gdk}	$C_{oxp}W_kL_D, \qquad k \in \{2,4\}$
C_{gs1}	$C_{oxn}W_1L_D + \frac{2}{3}C_{oxn}W_1L_1$
C_{gs4}	$C_{oxp}W_4L_D + \frac{2}{3}C_{oxp}W_4L_4$
C_{dw2}	$\frac{C_{JP}A_{D2}}{\left(1+V_{SD2}/\phi_{BP}\right)^{MJP}} + \frac{C_{JSWP}P_{D2}}{\left(1+V_{SD2}/\phi_{BP}\right)^{MJSWP}}$
C_{dw4}	$\frac{C_{JP}A_{D4}}{\left(1+V_{SD4}/\phi_{BP}\right)^{MJP}} + \frac{C_{JSWP}P_{D4}}{\left(1+V_{SD4}/\phi_{BP}\right)^{MJSWP}}$
C_{db1}	$\frac{C_{JN}A_{D1}}{\left(1+\left(V_1-V_{SS}\right)/\phi_{BN}\right)^{MJN}} + \frac{C_{JSWN}P_{D1}}{\left(1+\left(V_1-V_{SS}\right)/\phi_{BN}\right)^{MJSWN}}$
C_{db3}	$\frac{C_{JN}A_{D3}}{\left(1+V_{DS3}/\phi_{BN}\right)^{MJN}} + \frac{C_{JSWN}P_{D3}}{\left(1+V_{DS3}/\phi_{BN}\right)^{MJSWN}}$
C_{db5}	$\frac{C_{JN}A_{D5}}{\left(1+V_{DS5}/\phi_{BN}\right)^{MJN}} + \frac{C_{JSWN}P_{D5}}{\left(1+V_{DS5}/\phi_{BN}\right)^{MJSWN}}$
C_{sb1}	$\frac{C_{JN}A_{S1}}{\left(1+V_{DS3}/\phi_{BN}\right)^{MJN}} + \frac{C_{JSWN}P_{S1}}{\left(1+V_{DS3}/\phi_{BN}\right)^{MJSWN}}$

Open-Loop Response

The open-loop differential transfer function of (293) can be factorized into the form of (299) using the *rootpoly()* function.

$$H_{ol} = \frac{A_{ol} \cdot (1 - s/Z_1) \cdot (1 - s/Z_2) \cdot (1 - s/Z_3)}{(1 - s/P_{ol1}) \cdot (1 - s/P_{ol2}) \cdot (1 - s/P_{ol3})} \tag{299}$$

The locations of the poles and zeros are determined via (300)-(302).

$$\{P_{ol1}, P_{ol2}, P_{ol3}\} = rootpoly(b_0, b_1, b_2, b_3) \tag{300}$$

$$\{Z_1, Z_2\} = rootpoly(a_{10}, a_{11}, a_{12}) \tag{301}$$

$$Z_3 = \frac{-a_{20}}{a_{21}} = \frac{g_{m1}}{c_{gd1}} \tag{302}$$

The DC gain is computed by substituting *s*=0 in (293).

$$A_{ol} = \frac{a_{10} \cdot a_{20}}{b_0} = \frac{g_{m1} \cdot g_{m4}}{g_{dd} \cdot g_{do}} \tag{303}$$

The overall DC gain given in (303) is the product of the DC gains of the individual stages. The gains of the first and second stages are given in (304) and (305) respectively.

$$A_{ol1} = -\frac{g_{m1}}{g_{dd}} \tag{304}$$

$$A_{ol2} = -\frac{g_{m4}}{g_{do}} \tag{305}$$

Closed-Loop Response

If the two-stage amplifier of Fig. 67 is used as the amplifier in a feedback configuration, then the closed-loop transfer function can be found by substituting the numerator and denominator from (294) into (143).

$$H_{cl}(s) = \frac{n_3 s^3 + n_2 s^2 + n_1 s + n_0}{b_3 s^3 + b_2 s^2 + b_1 s + b_0 + \beta(n_3 s^3 + n_2 s^2 + n_1 s + n_0)} \tag{306}$$

Which can also be expressed as:

$$H_{cl}(s) = \frac{n_3 s^3 + n_2 s^2 + n_1 s + n_0}{d_3 s^3 + d_2 s^2 + d_1 s + d_0} \tag{307}$$

where

$$d_3 = b_3 + \beta \cdot n_3 \tag{308}$$

$$d_2 = b_2 + \beta \cdot n_2 \tag{309}$$

$$d_1 = b_1 + \beta \cdot n_1 \tag{310}$$

$$d_0 = b_0 + \beta \cdot n_0 \tag{311}$$

The poles migrate as the feedback factor varies. For a given feedback factor, the closed-loop pole locations are given by:

$$\{P_{cl1}, P_{cl2}, P_{cl3}\} = rootpoly(d_0, d_1, d_2, d_3) \tag{312}$$

The closed-loop DC gain is computed by substituting s=0 in (307).

$$A_{cl} = \frac{n_0}{d_0} \tag{313}$$

To ensure that only stable closed-loop systems are considered, constraints were added to the design specification file to ensure that all the closed-loop poles lie in the left half-plane. Using the poles defined in (312), the zeros given in (301) and (302), and the DC gain given in (313), the transfer function of (307) can be expressed as:

$$H_{cl}(s) = \frac{A_{cl} \cdot (1 - s/Z_1) \cdot (1 - s/Z_2) \cdot (1 - s/Z_3)}{(1 - s/P_{cl1}) \cdot (1 - s/P_{cl2})(1 - s/P_{cl3})} \tag{314}$$

Linear Step Response

The closed-loop system's transient step response computed using the inverse Laplace Transform is given in (315).

$$v_{od}(t) = A_{cl} \cdot \xi \cdot \left\{1 + k_1 \cdot e^{P_{cl1} \cdot t} + k_2 \cdot e^{P_{cl2} \cdot t} + k_3 \cdot e^{P_{cl3} \cdot t}\right\} \tag{315}$$

The variable ξ represents the magnitude of the input step and the coefficients of the exponentials are given by (316)-(318).

$$k_1 = -\frac{(1 - P_{cl1}/Z_1)(1 - P_{cl1}/Z_2)(1 - P_{cl1}/Z_3)}{(1 - P_{cl1}/P_{cl2})(1 - P_{cl1}/P_{cl3})} \tag{316}$$

$$k_2 = -\frac{(1 - P_{cl2}/Z_1)(1 - P_{cl2}/Z_2)(1 - P_{cl2}/Z_3)}{(1 - P_{cl2}/P_{cl1})(1 - P_{cl2}/P_{cl3})} \tag{317}$$

$$k_3 = -\frac{(1 - P_{cl3}/Z_1)(1 - P_{cl3}/Z_2)(1 - P_{cl3}/Z_3)}{(1 - P_{cl3}/P_{cl1})(1 - P_{cl3}/P_{cl2})} \tag{318}$$

Due to nonlinearity, the real transient response and (315) will not exactly agree. For example, for large input steps, the amplifier may initially slew toward the final value. Furthermore, as the node voltages and branch currents deviate from their quiescent values with signal swing, the small-signal parameters of the devices vary. Despite the fact that the response of (315) is inexact, designers still find it useful to look at these expressions nonetheless.

Bibliography

[44] B.Y. Kamath, R. G. Meyer, and P. R. Gray, "Relationship Between Frequency Response and Settling Time of Operational Amplifiers," *IEEE J. Solid-State Circuits,* vol. SC-9, pp. 347-352, Dec. 1974.

[45] Y. Chen, M. E. Schlarmann, and R. L. Geiger, "An Improved Design Formulation for Design and Optimization of Operational Amplifiers," *Proc. MWSCAS 1999*, Las Cruces, New Mexico, Aug 8-11, 1999, pp. 72-75

CHAPTER 7. A SIMPLE CMOS TRANSRESISTOR

A paper published in IEE Electronics Letters

Mark E. Schlarmann and Randall L. Geiger

Abstract

A simple highly linear transresistor is reported. Simulation and experimental results are presented.

Introduction:

Due to the nonlinear I-V characteristics of the MOS transistor, distortion is a serious problem in MOS resistive circuits (MRC's). As a result, several linearization techniques for MRC's have emerged. Most reported techniques have been derived by inspection of simple analytical device models to deduce the conditions under which full or partial suppression of nonlinear terms takes place (e.g. [46, 47]). In practice, these circuits do not perform as well as predicted because the simple analytical models used in their development do not accurately model the distortion characteristics of a MOS transistor [48-50]. With two notable exceptions [51, 52], neglecting mobility variation appears to be the dominant source of modeling error contributing to the residual nonlinearities [49, 50]. In this letter we report a simple and compact transresistor with linearity characteristics that compare favorably with the most linear, higher-complexity transresistors previously reported.

Proposed Circuit:

The proposed transresistor is shown in Fig. 1a. I_{BIAS} is large enough to ensure that M2 is strongly inverted and saturated. M2 bootstraps the gate voltage of M1 to its drain. The offset provided by the bootstrap circuit is larger than the threshold voltage of M1 ensuring that for non-negative input currents M1 operates in the linear region. I_{IN} develops a voltage V_1 at the drain of M1. The relationship between I_{IN} and V_1 is highly linear. Bootstrapping V_O to V_1, in turn, establishes a linear relationship between I_{IN} and V_O. The linearity is actually improved by the imperfect level-shift circuit. The channel conductance of M2 causes a slight fluctuation in the gate-drain voltage of M1. This variation partially offsets the nonlinearities.

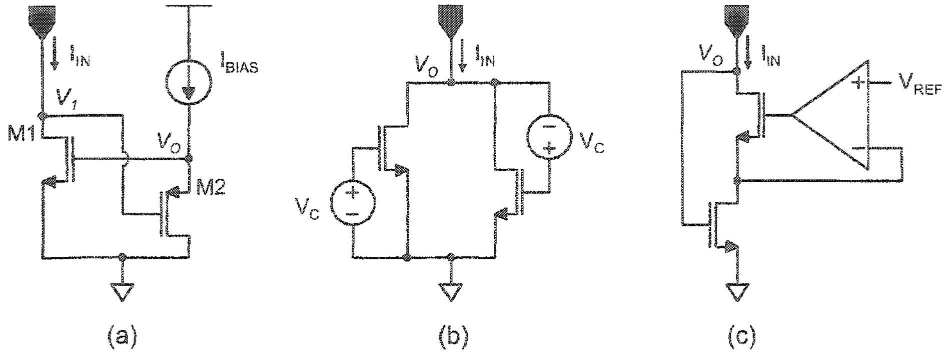


Fig. 71 Single-ended transresistors (a) proposed, (b) Banu, and (c) Wyszynski

Assuming a high-impedance current source, the transresistance gain is approximately:

$$R_T = \left(\frac{1}{g_{m1} + g_{o1}} \right)\Bigg|_{Q-Pt.} = \frac{L_1}{\mu_n C_{ox} W_1 \left(V_{1Q} - V_{T1Q} \right)} \tag{319}$$

where g_{m1} and g_{o1} are the small-signal model parameters of M1, V_{1Q} is the quiescent value of V_1, and V_{T1Q} is threshold voltage of M1 at the quiescent point.

Simulation Results

Two of the most linear single-ended transresistors reported are included in Fig. 1. The structure of Fig. 1b from Banu [46] achieves partial cancellation of even-ordered nonlinearities in the absence of mobility degradation. The transconductor by Wyszinski [53] configured as a transresistor is shown in Fig. 1c. The latter circuit requires an amplifier but if the often used, symmetric models are used for the MOSFETs, a linear transresistance is obtained.

The transresistors of Fig. 1 were each designed to achieve a nominal transresistance of 10KΩ in a 0.25μ 2.5V CMOS process. Simulations were performed with HSPICE using a BSIM3 (Level 49) model. A single p-channel transistor was used to provide I_{BIAS} for the proposed circuit and a linear single-pole macro-model was used for the amplifier of Fig. 1c. A sinusoidal input current superimposed on a quiescent current of 25uA was applied at the inputs. Fig. 2 shows a comparison of the total harmonic distortion (THD) as a function of the output swing. From these results, it is apparent that the proposed transresistor is substantially more linear than the existing structures for signal swings up to 1 V_{P-P}.

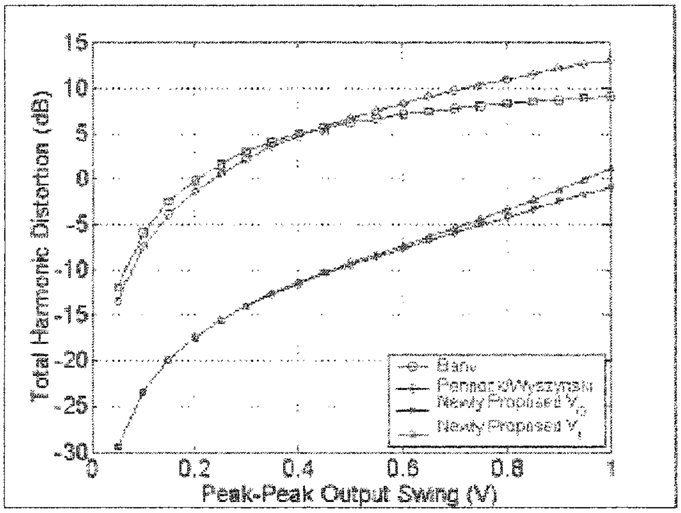


Fig. 72. Total harmonic distortion for single-ended structures

Balanced versions of the transresistors of Fig. 1 were simulated by exciting two single-ended transresistors differentially. The quiescent operating points and the nominal transresistances were the same as used for the single-ended simulations. The simulations show that the THD is reduced by more than 40dB via the use of balanced structures for differential swings up to 2V. The linearity characteristics of the proposed circuit and the Wyszynski circuit are nearly identical. Their THDs differ by less than 0.7 dB over the specified signal range. These two circuits exhibit at least a 6 dB reduction in THD over the output signal swing range when compared with the Banu structure.

Application in a Transresistance Gain Stage:

The transresistors of Fig. 1 are not suitable for driving resistive loads because they do not have low output impedances. A low output impedance transresistor can be realized by putting a transconductor in the feedback path of an op-amp as shown in differential form in Fig. 3a. This circuit was simulated using the subcircuits of Figures 3b and 3c to realize the resistive feedback network. Fig. 3b shows a balanced version of the proposed transresistor configured as a differential-in/differential-out transconductor. Fig. 3c shows the popular Czarnul/Song [47] transconductor. Using simple analytical models for the MOS transistors, both even- and odd-ordered nonlinearities are cancelled. Practically, mobility degradation limits the performance of the circuit.

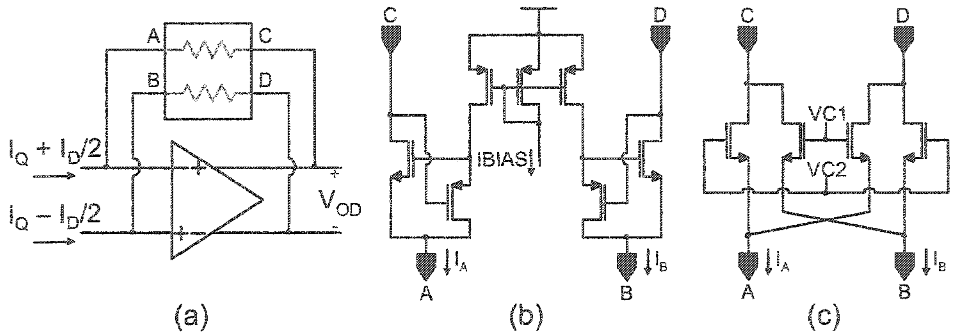


Fig. 73. Application as a transresistance gain stage (a) Gain stage using a transconductance feedback network, (b) balanced version of the proposed circuit, and (c) Czarnul/Song circuit

Fig. 4 shows the simulated THD as a function of the output swing for low-frequency sinusoidal differential input currents ranging from 5μA to 80μA. These simulations suggest the new circuit offers a significant improvement in linearity compared to the Czarnul/Song circuit.

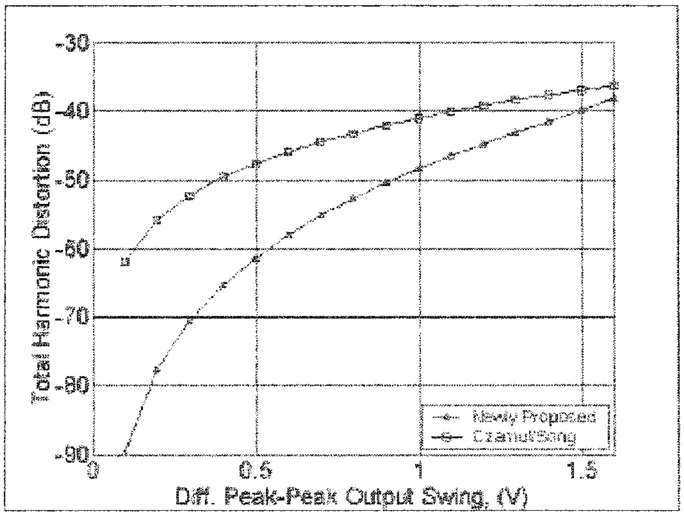


Fig. 74. Total harmonic distortion of the transresistance gain stage for two different feedback networks

Experimental Results:

The proposed circuit was constructed using a transistor array fabricated in a 5μ CMOS process. Since the devices were prefabricated, no attempt was made to optimize the individual or relative device sizes. Device dimensions were W1/L1 = 42/5 and W2/L2 = 42/7 in μm. The circuit was tested at supply voltages of 5 and 10V. For the 5V case, the transresistance gain was 2.5 kΩ. The max. deviation from linear relative to the full-scale deflection was ±0.35% for an output swing of 1.5V. For the 10V case, the transresistance gain was 1.6 kΩ. The max. deviation from linear was ±0.4% for an output swing of 5V. Even better performance could be achieved by optimizing the device sizes and/or by implementing balanced versions of the circuit.

Conclusions:

A simple, compact transresistor was introduced. Simulations show its linearity properties are attractive when compared to some of the best transresistors reported in the literature. Experimental results from a single-ended structure indicate a linearity deviation bound by ±0.35% over a voltage swing of 1.5 Vp-p. Balanced versions of the circuit are even more linear.

Acknowledgment:

Support for this project was provided, in part, by Texas Instruments Inc., National Semiconductor Inc., Dallas Semiconductor Inc., and the R. J. Carver Charitable Trust.

References:

[46] Banu, M. and Tsividis, Y.: 'Floating Voltage-Controlled Resistors in CMOS Technology', Electron. Lett., 1982, 18, (15), pp. 678-679

[47] Czarnul, Z.: 'Modification of Banu-Tsividis Continuous-Time Integrator Structure', IEEE Trans. Circuits Syst., vol. CAS-33, pp. 714-716, July. 1986

[48] Flandre, D. and Jespers, P.: 'Charge-Sheet Modelling of MOS I-V Fundamental Nonlinearities in MOSFET-C Continuous-Time Filters', Electron. Lett., 1995, 31, (17), pp. 1419-1420

[49] Groenewold, G. and Lubbers, W. J.: 'Systematic Distortion Analysis for MOSFET Integrators with Use of a New MOSFET Model', IEEE Trans. Circuits Syst. II, vol. 41, pp. 569-580, Sept. 1994

[50] Vidal, E., Martínez, H., Alarcón, E. and Poveda, A.: 'Nonlinear Analytical Model of the MRC (MOS Resistive Circuit)', in *Proc. 1999 Midwest Symposium on Circuits and Systems*, Las Cruces, New Mexico

[51] Tsividis, Y. and Vavelidis, K.: 'Linear, Electronically Tunable Resistor', Electron. Lett., 1992, 28, (25), pp. 2303-2305

[52] Shoucair, F. S., and Patterson, W. R., "Analysis and Modeling of Nonlinearities in VLSI MOSFET's Including Substrate Effects," *IEEE J. Solid-State Circuits*, 1993, vol. 40, no. 10, pp. 1760-1767

[53] Wyszynski, A.: 'Low-Voltage CMOS and BiCMOS Triode Transconductors and Integrators with Gain-Enhanced Linearity and Output Impedance', Electron. Lett., 1994, 30, (3), pp. 211-212

CHAPTER 8. CONCLUSION

This dissertation is a compilation of papers related to three different topics. The remainder of this chapter briefly summarizes the work that has been done and lists the basic contribution(s) associated with each topic.

Multistage Multipath Compensated Amplifier and Related Dipole Mismatch Tuning Technique

In order to truly realize entire mixed-signal systems on a chip and leverage the benefits of advanced process technologies, new low-voltage compatible amplifier topologies need to be developed.

In this context, a multipath-compensated multistage amplifier was introduced. These structures are compatible with low-voltage supplies because they use horizontal techniques (cascading) rather than vertical techniques (device stacking) to achieve large DC gains. When properly designed, these amplifiers are inherently first-order and do not suffer a reduction in the achievable gain-bandwidth product due to the process of compensation.

The technique relies upon pole-zero cancellation for proper operation. Absent techniques that ensure accurate pole-zero cancellations, these architectures are not practical for high-speed applications. This is due to the fact that imperfect cancellations result in the appearance of slow-settling components in the transient response. To overcome this problem, structures that inherently ensure accurate cancellation or those that tune themselves to compensate for variations need to be developed.

A tuning strategy for a two-stage multipath-compensated amplifier was developed. It is based upon the observation that if the low-frequency pole leads the zero, the step-response is underdamped. Conversely, if the zero leads the pole, it is overdamped. By sensing the slope of the transient step response after a certain delay, the relationship between the location of the pole and the zero can be determined. Utilizing this information, a bias current is adaptively adjusted to modify the location of the pole relative to the zero. The process is repeated many times driving the pole-zero mismatch down to an acceptable level. The concept was experimentally verified using a prototype fabricated in a 0.25μ CMOS process.

The difficulty of tuning an amplifier's dipole mismatches compounds with the number of stages in the amplifier. The insight gained in developing a tuning strategy for the two-stage amplifier has led to a methodology for tuning an amplifier with three or more stages. Preliminary simulations predict the technique is viable. Hardware implementation will probably require a digital signal processor in the tuning loop.

Contributions

The author's contributions associated with this project are:

- Developed a compensation technique for a multistage amplifier that does not sacrifice the GBW product as additional stages are cascaded.
- Technique for sensing and tuning dipole mismatch in a two-stage multipath-compensated amplifier was conceived, implemented, and experimentally proven.
- A method for solving the more difficult problem of tuning the dipoles' mismatches in a multipath-compensated amplifier composed of three or more stages was proposed and simulated.

Design Space Explorer

Design optimization tools that eliminate human interaction altogether may impede the acquisition of designer knowledge and thus delay or stall the progression to new or improved circuit topologies.

Using an equations-based approach to explore a design space is one way that designers can gain valuable insight into the available design tradeoffs. The enhanced understanding gained by the designer will result in better design realizations and might accelerate the progression to improved topologies in the future

Except in the simplest of cases, exploring a design space by hand is impractical. To address this issue, an interactive, network-centric circuit design tool and design knowledge repository has been developed. They allow a designer to interactively explore a circuit design space of pre-characterized circuit topologies using a convenient graphical user interface. Users can extend the system to include new or custom circuit topologies by writing their own design specification files.

Contributions

The author's contributions associated with this project are:

- *Design Space Explorer (DSE)* – an interactive CAD tool used to explore a design space. Published on the World-Wide-Web for others to use.

- DSE models for the fully-differential, the telescopic-cascode, and the two-stage amplifiers.

MOS Transresistor

Due to their large area requirements, polysilicon resistors are not an attractive option for some integrated applications. A new transresistor was developed that offers a more area-efficient way to convert signal currents into signal voltages. The new structure is very simple. It consists of two transistors and a current source. Simulations predict that the linearity of the new structure compares favorably with other transresistors that have reported in the literature. Yet, the complexity of the new structure is very low in comparison.

Contribution

The author's contributions associated with this project are:

- A new transresistor that exhibits reasonable linearity yet offers a very compact realization.

APPENDIX 1. ACCOMPANYING CD-ROM AND OPERATING INSTRUCTIONS

The accompanying CD-ROM contains a copy of the Design Space Explorer source code and documentation. Although it should be possible to inspect the source code on any machine capable of reading an ISO compatible CD-ROM, to compile and execute it will require Sun Microsystem's Java[1] Compiler and Java Runtime Environment. The code has been compiled and tested compatible with version 1.4 of the Java Development Kit which is freely available on the web at http://java.sun.com.

[1] Java is a registered Trademark of Sun Microsystems Inc.