Operational Amplifier Compilation with Performance Optimization

Hidetoshi Onodera, Hiroyuki Kanbara, and Keikichi Tamaru

Department of Electronics, Kyoto University
Sakyo-ku, Kyoto 606 Japan

ABSTRACT

We describe a new design methodology for analog circuits in which topological design is followed by simultaneous device sizing and layout design. By merging circuit and layout design into a single design process, analog circuits can be optimally designed taking layout parasitics fully into account. Based on the methodology, a CMOS operational amplifier compiler, OAC, has been developed. Given a set of performance specifications and process parameters, OAC generates a layout with circuit performance optimized to meet specified performance constraints. A new procedural layout technique is employed for generating compact and practical layouts. A non-linear optimization method is applied for device sizing which relies on the results of simulations based on the circuit extracted from the layout. Design experiments have shown that OAC can produce satisfactory results with respect to both circuit performance and layout density.

1. Introduction

There is growing demand for integrating analog circuitry such as A/D converters and filters on the same chip with digital circuitry. In most mixed analog/digital circuits, analog circuitry takes only a small portion of the chip area. However, these circuits require considerable effort with respect to their design cost. This stems from low design efficiency due to the varied objectives and constraints being considered in the design process. For lack of appropriate computer-aided design tools for analog circuits, the design of custom analog circuits still requires expert designers.

In various analog circuits, an operational amplifier is one of the most frequently used functional blocks. Therefore, the design automation of an operational amplifier would greatly reduce the design time and cost of analog circuits. Also, the method would be applicable to other basic functional blocks and hence form a basis for automating the design of higher-level functional blocks.

From the above reasons, several results have been reported on the design automation of analog circuits, especially with operational amplifier synthesis as the target[1-5]. They produce circuit schematics including transistor sizes and component values from performance and process specifications. However, some performance parameters, such as phase margin and unity gain frequency, are strongly influenced by circuit parasitics which can be evaluated only after the completion of layout design. Therefore, even if a sized schematic is automatically obtained, there still remains intensive labor first in layout design, and then in design modification in order to compensate for the parasitic effects. In order for an automated design system to be more practical and reliable, it should include layout directly in the design cycle and guarantee the performance of synthesized circuits by simulation results based on extracted layout.

Recently, some automatic synthesis tools have accommodated such needs by providing a complete path from specification to layout[6-8]. However, they rely on a conventional design flow in which circuit design and layout design take place sequentially. Only an iterative feedback path from layout design to circuit design is provided, hence design modification for compensating layout parasitics is a long process.

In this paper, we propose a new design methodology for basic analog functional blocks. The design process consists of global and detailed design in contrast to the sequential circuit and layout design used in the conventional approach. For global design, a circuit topology is first determined and rough device sizing is processed. In the detailed design phase, device sizing and layout design are carried out simultaneously thereby taking parasitic effects into account precisely. Based on this methodology, a CMOS operational amplifier compiler, called OAC, has been developed. Given a set of performance specifications and process parameters, OAC produces a layout whose circuit performance is optimized under specified performance constraints. A new procedural layout technique is employed for generating compact and practical layouts. A non-linear optimization technique is applied to the design optimization process in which circuit performance is optimized taking the resulting layout into account. Design experiments have shown that OAC can produce satisfactory results with respect to both circuit performance and layout.

The paper is organized as follows. Section 2 describes the proposed design methodology for basic analog functional blocks. Section 3 gives an overview of OAC. Section 4 describes the method for topology selection and device sizing of the global design. Section 5 treats details of the design which includes the methods used for design optimization and procedural layout description. Section 6 shows some experimental results. Finally, Section 7 presents concluding remarks.

2. Analog Design Methodology

In this section, we focus on a design methodology suitable for basic analog functional blocks such as operational amplifiers, comparators, and other similar blocks. First we examine a design process performed by a human designer and then consider a method for automating the process.

A design flow followed by a human designer can be expressed as Fig. 1. Given a set of design specifications, first a

IEEE 1989 CUSTOM INTEGRATED CIRCUITS CONFERENCE

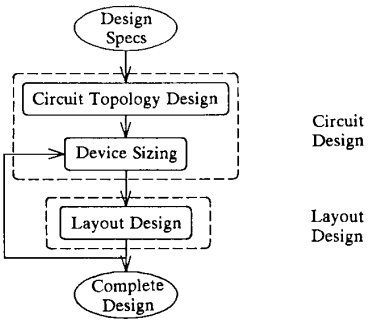


Fig. 1 Conventional design flow.

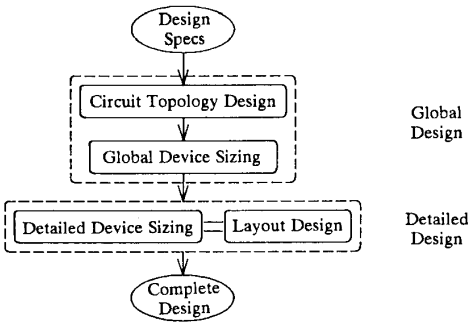


Fig. 2 Proposed design flow.

designer determines a circuit topology. It is most frequently the case that a known topology is modified as opposed to creating a completely new one. Usually it is selected from well characterized topologies or previously adopted ones, or built by combining them. Decision criteria rely on rough performance estimations. Next, the designable parameters of a selected topology, such as transistor sizes and component values, are determined so that given specifications are satisfied. Early in this stage, a designer may use analytical equations which express the relations between circuit performance and design parameters, and then assign approximate values. Later these values are refined according to accurate performance evaluations performed by a circuit simulator such as SPICE. Layout design begins after the completion of circuit design. In analog circuits, the geometry of devices and layout has a strong influence on certain performance parameters. For instance, the offset voltage of a differential amplifier is related to the degree of symmetry of the input devices. Also, frequency characteristics are affected by layout parasitics. Therefore, various constrains should be taken into account in the layout process. After layout extraction and simulation, circuit performance is often different from that estimated before the layout design. If the discrepancy is permissible, the design process is finished. If not, going back to the circuit design stage, values of design parameters are modified. This modification is continued until the desired performance goals are attained. By its nature, the process shown in Fig. 1 is sequential and iterative.

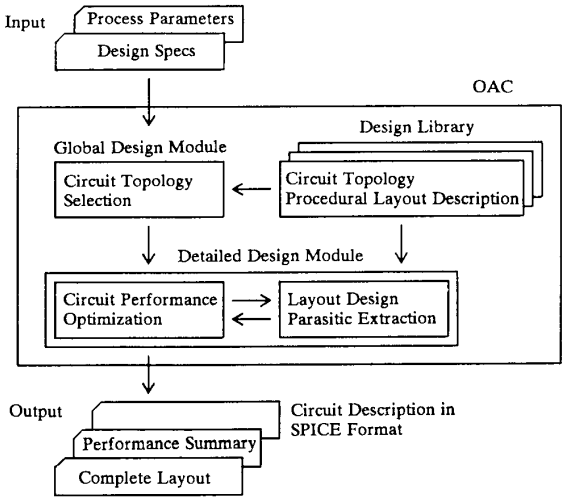


Fig. 3 Organization of OAC (Operational Amplifier Compiler).

As we have seen, it can be said that a human designer does not complete a design part by part, but rather refines the capturing of circuit performance goals on a global basis. The design process is guided by performance evaluation with accuracy comparable to that of each design level. However, the situation in which design modifications are required after layout design indicates that the accuracy of the performance evaluations is not sufficient for the determination of design parameters unless layout information is provided.

Therefore, we introduce the following principle for automated design of analog circuits: *a circuit is designed step-by-step starting from a global level to the detailed level according to performance evaluations with accuracy comparable to that of each design level.* As shown in Fig. 2, device sizing and layout are coupled and under direct control of the global estimation process; the sequential iterative problems shown in Fig. 1 are thereby avoided. In global design, first a circuit topology appropriate to the given specifications is determined and then approximate values are assigned to each of the design parameters. The detailed analysis, layout and the refinement of parameters phases are carried out simultaneously in order to optimize the design guided by accurate performance evaluations based on extracted layouts.

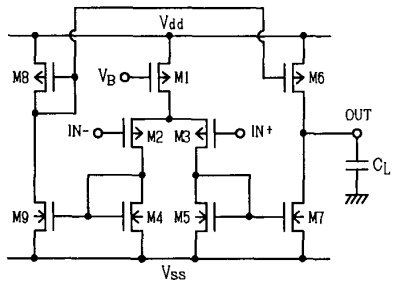
3. OAC (Operational Amplifier Compiler): Overview

The above methodology has been implemented in a CMOS operational amplifier compiler, OAC, whose conceptual structure is shown in Fig. 3. The input of OAC consists of

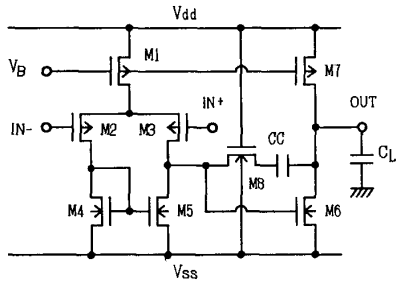
- a set of performance specifications,
- a set of process parameters (MOS model parameters for SPICE and layout design rules), and
- constraints on cell dimensions (optional).

OAC produces

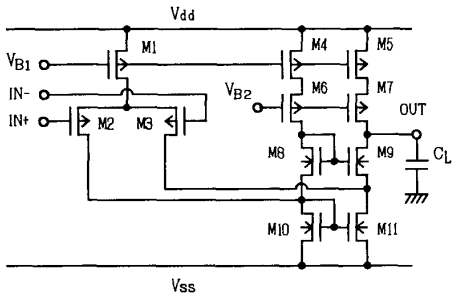
- a complete layout of an optimized circuit in CIF,
- a circuit description in SPICE format, and
- a performance summary.



(a) A one-stage circuit for low frequency applications.



(b) A conventional two-stage circuit.



(c) A folded cascode circuit for high frequency applications.

Fig. 4 Circuit topologies in the library.

OAC consists of global and detailed design modules as well as a design library. The detailed design module is further divided into a circuit performance optimizing module utilizing a non-linear optimization technique and a layout design and parameter extraction module.

The design library stores circuit topologies with their layout descriptions. Currently, OAC can deal with three circuit topologies shown in Fig. 4. They are: a one-stage amplifier for low power, low frequency applications [Fig. 4(a)], a basic two-stage amplifier [Fig. 4(b)], and a folded-cascode amplifier for high frequency applications [(Fig. 4(c)]. For each circuit, we assume that the operating conditions such as supply voltages *Vdd Vss*, the bias voltage *Vb*, and the load capacitance *Cl* are given together with performance specifications, whereas each transistor size and the feedback capacitance *Cc* are considered as design parameters to be determined.

The processing flow in OAC is outlined in the following. First, the global design module selects the most appropriate circuit topology for a given set of specifications from the design library. Then, it assigns an approximate value to each design parameter and passes it to the detailed design module. The detailed design module determines each design parameter by applying a non-linear optimization method under the given performance constraints. Every time a trial design is created in the optimization process, the corresponding layout is generated according to the procedural layout description for extracting the parasitics. These values are used in the ensuing simulation. Hence, the circuit performance is optimized under the given constraints taking into account the circuit parasitics. In the layout generation phase, the area is minimized by adjusting the number of gate folds with/without a cell-height constraint.

Except for the parameter assignment portion in the global design module, OAC is written in C and FORTRAN and runs under UTS® on a FACOM M780 mainframe and also under UNIX® on a Sun 3 workstation.

4. Global Design

The objective of the global design is to select a circuit topology and assign approximate values to each design parameter for a given set of performance specifications. The basic strategy employed for the problem is to select and modify a proper design example from the design library.

For that purpose, we store several sets of design parameter values for each circuit topology in the design library together with their operating conditions and circuit performances. Since circuit performance depends on the operating conditions and process parameters as well as the design parameters, first performance parameters which represent the circuit performance are calibrated for specified operating conditions and process parameters. Then, the calibrated performance parameters for each design example are compared to given specifications and rated concerning for the ability to achieve those goals. We use a six-level rating from 1 (very easy) to 6 (out of reach). The rating is provided in order to compensate for the differences in the feasible region of each performance parameter for each circuit topology. According to the results, a design example is selected that is estimated to have the highest possibility for meeting the specifications.

Next, starting from the selected design example, the design parameters are modified step-by-step for improving the worst performance parameter according to design plans in the knowledge-base. The modification process is continued until all the ratings become less than or equal to 3 (average) or unless further improvement cannot become achieved.

5. Detailed Design

5.1. Design Optimization

During the detailed design, all design parameters are determined such that the circuit performance is optimized for a given set of performance specifications taking the resulting layout into consideration. This problem can be formulated as a multi-objective optimization problem[5,9]. For that purpose, we define the following evaluating function for the *i*-the performance parameter.

$$f_i(\vec{x}) = W_i(\sqrt{(P_i(\vec{x})-1)^2 + (1/W_i)^2} \pm (P_i(\vec{x})-1)) \qquad (1)$$

where $\vec{x}$ represents a design parameter vector, $P_i(\vec{x})$ is the i-th performance parameter normalized by the corresponding design specification, and W_i is a weighting factor which represents a relative priority for $P_i(\vec{x})$. The plus (minus) sign is applied when $P_i(\vec{x})$ is required to be less (greater) than the specification. In the case $P_i(\vec{x})$ is requested to have the same value as the specification, only the square root term is used for evaluating the function. The function imposes a kind of outer penalty on the design with respect to the given performance specification. When the circuit performance does not satisfy the specification, it gives a high penalty, approximately proportional to the amount of under-attainment. When the circuit attains the specification, the value of the function becomes unity and further decreases as the circuit "over-attains" its goal. Using the function, the optimal design problem can be defined as

$$\text{minimize} \quad \frac{1}{2}\sum_{i=1}^{m} f_i(\vec{x})^2 \qquad \vec{x} \in E^n \qquad (2)$$

where n is the number of design parameters and m is that of performance parameters considered for optimization.

Many non-linear optimization techniques are applicable to the problem. Here, we adopt the Levenbarg-Marquardt method[10] because of its high convergence speed. In order to stabilize the optimization process, constraints on the feasible regions for the design parameters are imposed by a variable transformation method[11].

During the optimization, the selection of independent design parameters is quite important. Ideally, we should consider all the design parameters as independent variables and search for an optimum solution. However, the search space becomes explosively large and its surface tends to become ill-behaved, which makes the search process inefficient. Therefore, we should impose proper constraints on the design parameters and choose only essential parameters for independent variables. In case of the basic two-stage circuit [Fig. 4(a)], for example, the *W/L* ratio of M8 should be determined in such a way that the channel resistance of M8 becomes equal to the inverse of the transconductance of M6 in order to cancel the right-half plane zero in the transfer function. In addition, there are several relations which should hold between certain *L*, *W*, and *W/L* ratios, which help to make the search space smooth and the optimization process reliable. These constraints are part of the design knowledge which human experts usually rely on and can be derived from the understanding of the operating principles of the circuit.

Performance evaluation in the detailed design stage should be accurate enough for guiding the design parameters to their optimal values. Basically we employ SPICE simulations in which parasitics of the corresponding layout are taken into account. However, in order to reduce computation time, only DC and AC analysis are performed. A performance parameter which is not directly obtained from the analysis, such as slew rate, is estimated from the operating point and small signal information.

Owing to the modular structure of OAC, circuit characteristics considered for optimization and their evaluation procedures can be modified easily.

5.2. Layout Design

In considering the automated design method of custom analog layouts, the following points should be taken into account.

1) There exist analog components with complex structure of widely varying sizes.
2) Complex topological and geometrical constraints, such as requirements for symmetry in routing as well as placement, are imposed on each layout.

Besides these, since layouts are repeatedly generated and evaluated in the proposed methodology, the following two points are crucial.

3) Layout parasitics can be easily extracted.
4) Computational cost should be low.

One possible way is to adopt auto-place and route strategies[6,7]. This approach offers flexibility in treating a wide variety of circuit topologies. However, it is still difficult to produce a compact and practical layout in a short period of computation time while satisfying imposed constraints.

Another promising way is to use procedural layout techniques[12-14]. Complex constraints can be expressed directly in procedural descriptions. The method for parasitic extraction can be described procedurally. Also, the computation time is expected to be short because layouts can be generated simply by following procedural descriptions. However, previously reported techniques are not well suited for analog layout generation in regard to the first item described above. Therefore, not only for managing size differences but also for taking advantage of the degrees of freedom in deciding the aspect ratios and terminal positions of large devices, we have developed layout description language in the form of special layout procedures using the C language and built as a layout design module which searches for an optimal solution.

The layout language has the following features.

- A layout primitive is a rectangle (also called a *box*). A layout is described hierarchically as a collection of boxes.
- A terminal is defined as a line segment (of course a point is permissible). This enables us to express degrees of freedom in terminal locations for large devices.
- All boxes and terminals have individual *names*, through which the design hierarchy is reflected. For example, box *name1* in cell *name2* has name *name2.name1*.
- A set of functions which support the description of layout is provided. It includes functions which: give coordinate values of specified boxes, give end points to be connected along two terminal segments, and extract parasitics associated with specified boxes. Boxes and terminals are referenced by their names.

Besides the support functions, functions are provided for describing various layout components from a box to analog macros such as differential pairs and current mirrors. Using these functions, a designer can describe a custom analog layout hierarchically in a parameterized form with respect to design parameters as well as layout design rules.

For example, the following is a description for placing a differential pair M2 M3, placing an interdigitated transistor M1, and connecting the source of M2 M3 with the drain of M1.

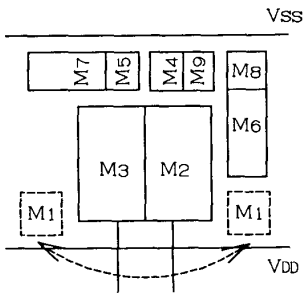


Fig. 5 Layout plan of the one-stage circuit. Transistor M1 is placed on either side indicated by the arrow.

```
diff_pair_("m2m3", &tr[2], &tr[3]);
place_("m2m3", x, y, ori);
rx = x_right_(PDIFF, "m2m3");
comb_tr_("m1", &tr[1]);
place_("m1", (rx+dist), y, ori);
h_v_connect_terminal_(AL1, "m2m3.source", AL1,
                      "m1.drain", W2, AL1, "bias");
```

The first two statements declare a differential pair given instance name *m2m3* and place it at (x, y) in the orientation denoted by *ori*. Information on transistors (L, W, etc.) is given by pointers *tr[ ]*. The third statement obtains the rightmost position *rx* of the p-diffusion of *m2m3*. The fourth and fifth statements declare and place an interdigitated transistor given name *m1*. The sixth statement connects the source terminal of *m2m3* (*m2m3.source*) and the drain terminal of *m1* (*m1.drain*) by the first metal wire and assigns it the name *bias*. Since terminals are defined as line segments, the nearest two points are connected in a "horizontal-vertical" manner. The parasitic capacitance to the substrate associated with the wire can be obtained by the support function:

```
cp = c_parasitic_(AL1, "bias");
```

As we have seen, the extraction of coordinate values and parasitics are easily performed through cross reference to layout elements by names.

Actual layout design starts from drawing a layout plan which indicates a topological relationship between components. Then, according to the plan, a layout description is made. A layout plan is also part of the design knowledge base. It should involve imposed constraints as well as layout heuristics suggested by the designer. Figure 5 is an example of a layout plan corresponding to the circuit shown in Fig. 4(a). Preservation of symmetry in the differential pair, reduction of parasitics which contribute to the second pole, and minimization of layout area are taken into account.

Parameterization of a layout includes the number of gate folds and the placement positions of certain components. When a layout is generated, the layout design module determines the values of these parameters so that the layout area is minimized. Since these parameters take discrete values, a minimum area layout is explored by evaluating the layout area of every feasible set of parameter values. If constraints on cell height are given, the layout design module selects a minimum area layout from the ones which do not exceed the specified cell height and then modifies that layout according to the constraints.

6. Design Examples

This section describes some examples and results of operational amplifier compilation performed by OAC. We employ device parameters and layout design rules from an industrial 2 μm double-metal CMOS process. Since the parameter assignment part of the global design module is currently under development, detailed design starts from the design parameter values of a selected design example.

A number of experiments under various performance specifications have revealed that in many cases OAC successfully attains proper designs if the given specifications are reasonable. The proper constraints in the form of relations between certain design parameters contribute to the reliability of OAC. Table 1 lists three sets of performance specifications (Specs A, B, C) and the resulting performance parameters for the compiled circuits. The corresponding layouts are shown in Fig. 6. Specification A is a demanding one since it is set close to the circuit performance of a manually designed circuit. Nevertheless OAC has successfully reached a solution which meets all the specifications. As for the other specifications, satisfactory results have been obtained.

The CPU times for the examples range from 54 to 127 seconds on a M 780 and 15 to 45 minutes on a Sun 3. In the compilation processes, the layout design takes roughly 20 % of the total CPU time. The rest is consumed in performing evaluations based on SPICE simulations in the optimization process.

In order to evaluate the proposed design methodology in which circuit design and layout design are carried out simultaneously, we have compiled the above examples in two different ways. The first method ignores all the layout parasitics in the optimization process and generates layouts after fixing the values of the design parameters. The second method, which is intermediate between the proposed method and the first method, considers layout parasitics assuming unit gate folding of each transistor in the optimization process, which can greatly reduce the layout design time. In the first method, although the apparent circuit performance after the device sizing meets all the specifications, the performance evaluations based on layout extractions have revealed that the phase margin requirements are not satisfied for all the examples. In the second method, similar results are obtained for Specs A and B. For Spec C, however, the method fails to meet the power dissipation requirement since the optimization process trys to compensate for the over-estimated parasitics resulted from the assumption by increasing the current consumption. From these results, we can confirm the effectiveness of the proposed methodology, especially in designing circuits susceptible to parasitics.

7. Conclusion

We have presented a design methodology for analog building blocks in which topological design is followed by simultaneous device sizing and layout design. Based on the methodology, an efficient automatic design tool for CMOS operational amplifiers, OAC, has been developed. Given a set of performance specifications and process parameters, OAC first selects a circuit topology and then optimizes circuit performance under the given constraints taking parasitics of the resulting layout into account. The results of the design process include a complete layout of an optimized circuit as well as circuit parameters. Design experi-

ments have shown that OAC can produce satisfactory results with respect both to circuit performance and generated layouts. At present OAC deals with operational amplifiers. However, it can be applied to other analog building blocks such as wide band amplifiers and comparators.

References

[1] P. E. Allen and H. Nevárez-Lozano, "Automated Design of MOS Op Amps," *Proc. 1983 ISCAS*, pp. 1286-1289, 1983.

[2] R. J. Bowman and D. L. Lane, "A Knowledge-Based System for Analog Integrated Circuit Design," *Proc. 1985 ICCAD*, pp. 210-212, 1985.

[3] M. G. R. Degrauwe et. al., "IDAC: An Interactive Design Tool for Analog CMOS Circuits," *IEEE J. Solid-State Circuits*, vol. SC-22, no. 6, pp. 1106-1116, 1987.

[4] R. Harjani, R. A. Rutenbar, and L. R. Carley, "A Prototype Framework for Knowledge-Based Analog Circuit Synthesis," *Proc. 24th Design Automation Conf.*, pp. 42-49, 1987.

[5] H. Y. Koh, C. H. Séquin, and P. R. Gray, "Automatic Synthesis of Operational Amplifiers Based on Analytic Circuit Models," *Proc. 1987 ICCAD*, pp. 502-505, 1987.

[6] J. Rijmenants, T. Schwarts, J Litsios, and R Zinszner, "ILAC: An Automated Layout Tool for Analog CMOS Circuits," *Proc. 1988 CICC*, pp. 7.6.1-7.6.4, 1988.

[7] D. J. Garrod, R. A. Rutenbar, and L. R. Carley, "Automatic Layout of Custom Analog Cells in ANAGRAM," *Proc. 1988 ICCAD*, pp. 544-547, 1988.

[8] H. Y. Koh, C. H. Séquin, and P. R. Gray, "Automatic Layout Generation for CMOS Operational Amplifiers," *Proc. 1988 ICCAD*, pp. 548-551, 1988.

[9] H. Onodera, H. Kanbara, K. Tamaru, "Module Generation of a CMOS Op Amp Using a Non-linear Optimization Method," *Trans. IEICE Japan*, vol. E71, no. 10, pp. 947-949, 1988.

[10] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *SIAM J. Appl. Math.*, vol. 11, no. 2, pp. 431-441, 1963.

[11] M. J. Box, "A Comparison of Several Current Optimization Methods, and the Use of Transformations in Constrained Problems," *Comput. J.*, vol. 9, no. 1, pp. 67-77, 1966.

[12] J. Kuhn, "Analog Module Generators for Silicon Compilation," *VLSI Systems Design*, 1987 May 4, pp. 74-80, 1987.

[13] M. R. Burich, "Programming Language Makes Silicon Compilation a Tailored Affair," *Electron. Des.*, vol. 33, no. 29, pp. 135-142, 1985.

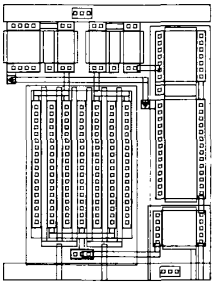
[14] W. E. Cory, "Layla: A VLSI Layout Language," *Proc. 22nd Design Automation Conf.*, pp. 245-251, 1985.

Table 1 Specifications and results for three examples.

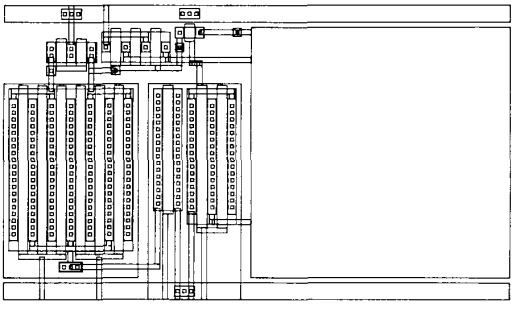
Parameter	Spec A	Result	Spec B	Result	Spec C	Result
DC Gain (dB)	70	70	80	84	55	57
Unity Gain Freq. (MHz)	0.50	0.50	3.0	3.0	240	240
Phase Margin (deg)	60	60	60	60	45	45
Slew Rate (V/μsec)	0.32	0.34	3.0	3.8	600	611
Noise ($nV/\sqrt{Hz}$)	90	64	200	191	300	287
Power Dissp. (mW)	0.022	0.021	0.50	0.48	15	15
Layout Area (μm^2)	minimize	123×165	minimize	303×174	minimize	216×212
Circuit Topology	-	(a)	-	(b)	-	(c)
Supply Voltage (V)	±2.5	←	±2.5	←	±2.5	←
Load Capacitance (pF)	10	←	10	←	1	←

[Below]

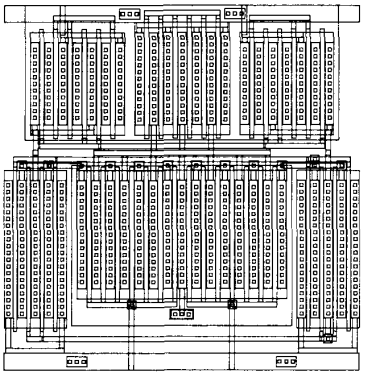
Fig. 6 Generated layouts for the examples. Figures (a), (b), and (c) correspond to Specs A, B, and C, respectively.



(a) One-stage circuit topology.



(b) Two-stage circuit topology.



(c) Folded cascode circuit topology.