Scientia Iranica, Vol. 6, No. 1, pp 59–73

SCIENTIA
IRANICA

Research Note

A Mixed Algorithmic and Knowledge-Based Approach for Automatic Design of Analog Circuits Based on a Behavioral Model

M. Shojaei[1] and M. Sharif-Bakhtiar*

In this paper, a novel method for automatic synthesis of analog circuits based on their behavior is presented. The provided method is a new mixed algorithmic and knowledge-based technique. In this method, the behavior and topological properties of the desired circuit is, first, derived. A behavioral graph with respect to the behavior of basic components is then devised to match the behavior of the desired circuit. The behavioral model is then transformed to a corresponding circuit for a possible solution. These operations utilize innovative algorithmic approaches for behavioral and topological modification based on KVL and KCL. Consequently, for a given design problem, this method is capable of producing various circuit topologies which do not necessarily exist in the library.

INTRODUCTION

Design of digital circuits is widely supported by sophisticated CAD tools. Hierarchical and structured abstractions are fully exploited to generate complex systems. On the contrary, much of the design of analog circuits is still hand-crafted by expert designers.

Economic factors, increasing demands for a wide variety of high performance analog circuits and growing requirements for single chip mixed designs have initiated efforts toward automating analog design. CAD tools specifically tailored to analog integrated circuit design promise to improve the design process in a wide variety of ways [1].

Analog CAD tools have not been developed to the extent of digital CAD tools due to inherent difficulties attributed to the analog design. Major difficulties in analog design automation are:

1. Complexity of analog design.
2. Inadequate or incomplete knowledge for the design of analog circuits.
3. Unavailability of topology construction for given specifications for a wide range of analog circuits.

For these reasons, topological design has been limited to selecting topologies from libraries in existing CAD tools [2-5].

In this article, a novel method for automatic synthesis of analog circuits based on a behavioral model is presented. This method handles the synthesis problems by performing design at the basic behavior level of the circuit for the automated design of analog circuits by mimicking human behavior. For a circuit with desired functions and specifications, the behavior is designed in such a way that the specifications are met and the structure of the circuit is simultaneously constructed with the use of behavioral and structural properties of KVL and KCL. Since KVL and KCL form the basis of topology construction, it is possible to derive new topologies from the basic topologies even at a low level as primitive as the level of individual elements. This will enable the design engine to devise complex circuits using a primitive and simple library which may only provide the basic electronic components.

The present method has been realized in an environment called BSDM (Behavioral based initial Synthesis, Diagnosis and Modification) which is also described in this paper. To enhance the speed and span of the design process, heuristic behavioral modeling for application of domain knowledge is also used in BSDM.

1. *Department of Electrical Engineering, Sharif University of Technology, P.O. Box 11365/9363, Tehran, I.R. Iran.*
*. *Corresponding Author, Department of Electrical Engineering, Sharif University of Technology, P.O. Box 11365/9363, Tehran, I.R. Iran.*

The application of heuristic modeling in conjunction with formal behavioral modeling based on KVL and KCL expands the design alternatives in BDSM.

A REVIEW OF KNOWLEDGE-BASED METHODS FOR AUTOMATIC DESIGN OF ANALOG CIRCUITS

In the past few years, there has been an intense interest in knowledge-based approaches to analog design automation [1-10]. Existing problems in analog circuit design automation were referred to in the previous section. These problems necessitate the use of knowledge-based approaches for automated analog design. Knowledge-based systems exploit a part of domain knowledge (heuristics, circuit architectures and design equations) which is not accessible to the formal methods to design analog integrated circuits. Surveys of knowledge-based analog design automation have been presented in a number of references [1,5,10].

The main design philosophy that has evolved and prevailed for unfixed topologies, so far, is the hierarchical approach [1]. The underlying idea involves the use of predesigned blocks stored in a library to partition the desired circuit into smaller distinct functional blocks connected to each other. This process is called topological design step. In the specification assignment step or sizing step of the design process, a set of specifications is assigned to each of the blocks chosen from the library. If these specifications are met, the performance of these blocks will yield the desired circuit performance. The same procedure is repeated in a similar manner for the smaller blocks at lower hierarchical levels. In order to carry out such partitioning of circuits and decomposition of specifications, a great deal of domain knowledge is required. This knowledge is mainly in the form of heuristics and design equations. Hierarchical design ensures the greatest design freedom among other approaches. Hierarchical systems are easier to expand and maintain, and they make better use of design knowledge [1,10]. The required library in hierarchical systems is also smaller than that of nonhierarchical methods [1]. Different variants of the basic hierarchical design approach have been realized [1,5-10], however, the following are still demanding problems:

1. Capture of expert basic knowledge.
2. Topology derivation instead of topology selection.
3. Handling interactive analog specifications.

OVERVIEW OF THE METHOD

In the existing hierarchical methods, different topologies are constructed from the configurations initially stored in the library. This causes the existing methods to be limited to decide only on the interconnection of the predefined circuit modules stored in the library. A large number of building blocks, along with their relevant data, are to be stored in the computer for the design of analog circuits. The stored data on the building blocks are to be changed in case of any variations in process parameters or the desired specifications. On the other hand, interaction of circuit modules makes it difficult to connect circuit blocks simply as in digital circuits. These limitations are due to the fact that the meaningful relation between the structure and the function of the circuit is not considered in the current methods. However, the function of a circuit (i.e., its purpose) is actually related to its structure (i.e., its schematic). Structure is what the circuit is and function is what the circuit is for, but behavior is what the circuit does [11]. Analog circuit behavior is characterized by the circuit equations and human designers use these equations at the abstract levels suitable for different design steps. Structural properties of the circuit are extracted from interconnection laws (KVL and KCL) present in circuit equations. This usage is a special feature of the design and modification process performed by human designers. Thus, to provide a formal model of expert designers' conception, circuit equations must be considered at the basic level of network equations. At this level, circuit variables will be currents and voltages of circuit modules.

Here, a novel concept is intoduced by considering the behavior of a circuit to interconnect the circuit modules by the use of KVL and KCL. (Automatic design of lumped circuits are considered.) This concept is, then, applied to topological design by expanding a primary topology of a circuit to a new topology in a formal manner. Consequently, this method is able to produce alternative topologies which are all potentially a solution to the given design problem.

The proposed concept provides a framework for the synthesis based on initial knowledge-based design and formal modification of circuit topology. To conduct this modification, the structural and behavioral failures are locally or globally located and identified in the structural and behavioral model of the circuit, respectively. These failures are, consequently, improved in a formal manner based on the relation between the behavior and the structure.

The purpose of the presented theory is:

- To concentrate on the circuit behavior as a radical solution for the current problems in the design of analog circuit topology.
- To provide a new formal framework for analog circuit design automation in which all aspects of analog circuit behavior can be gradually integrated.

The behavior of an analog circuit has different aspects which must be considered and modeled, such as:

- Functional behavior such as linearity or frequency behavior.
- Interaction of circuit modules' behavior which is essentially originated from interconnection laws (KVL and KCL).

In the presented method, different aspects of analog circuit behavior are considered and the way that circuit block diagram topology is formally constructed from circuit behavior without the use of library is demonstrated. The formal nature of the method makes it expandable to consider any other aspect of circuit behavior in future works.

In this method, behavioral modeling is done in its basic form, i.e., an abstract representation of circuit equations (KVL/KCL and input/output equations for circuit modules) [12]. The basic behavioral model is indeed a qualitative causal representation of the network equations. Network equations provide an appropriate representation of interaction between circuit modules according to the nature of the interconnections. Basic behavioral modeling provides the ability of comparing, composing and modifying basic interaction between circuit components. This modeling is hierarchical to retain a hierarchical design procedure.

To start the basic behavioral modeling, the following information is considered for circuits and domain knowledge:

- Structural information.
- Behavioral information including:
  - Causal relations between voltages/currents.
  - Qualitative constraints on causal relations and/or voltages/currents.

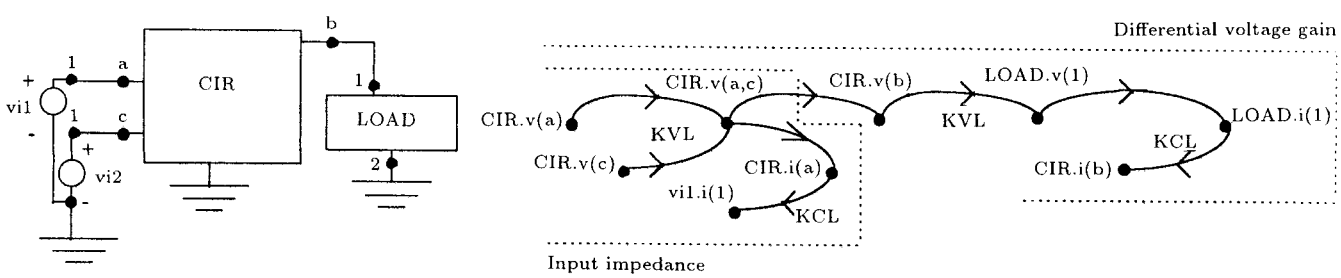
Structural information is information about the whole or some part of circuit architecture, which is known. To represent and manipulate circuit equations at the level of causal relations (i.e., generation of dependent variables from independent variables [11,13]), a graph, called here a B-graph (Behavioral graph), is defined to represent causal relations. A B-graph is a directed graph in which nodes denote terminal currents or terminal voltages or terminal pair voltages of circuit modules, whereas branches and paths represent causal relations between different quantities. The direction of each branch is from an independent quantity to the dependent quantities (Figure 1). In the notation used in Figure 1, for example, CIR.i(a) denotes electrical current of terminal "a" of circuit module CIR.

As shown in Figure 1, specific B-graphs are assigned to causal relations of behavior of LOAD and CIR modules as well as to KVL and KCL. "KVL" and "KCL" subgraphs model the behavioral interaction of CIR and LOAD.

A set of qualitative/quantitative constraints over causal relations, currents and voltages is assigned to each B-graph. Causal relations in the B-graph show what quantities interact and corresponding constraints represent the features of interaction. The use of qualitative constraints for specifications, causal relations and electrical quantities in the network equations provides the utility of applying qualitative reasoning for circuit behavior manipulation. Unlike numerical techniques, qualitative reasoning deals with the qualitative values and relationships between variables, rather than the actual values or the precise analytic equations [1]. It is, therefore, a powerful method for qualitatively locating and identifying failures in the network equations and manipulating the equations to modify behavior in a systematic way, all at the level of basic behavior.

Abstract level of causal relations and qualitative constraints does not depend on the linearity or nonlinearity of equations. Thus, a basic behavioral model is applied both for nonlinear and linear behavior of analog circuits.

In order to perform qualitative manipulations, a new discrete domain must be defined, which is called Quality Domain (QD) [1]. First, the considered continuous domain $A(A \subset \bar{R})$ is quantized to appropriate regions, open or closed, within which the quantitative variable behaves uniformly. This quantized version of the continuous domain, known as quantity space, is mapped to appropriate qualitative values in the quality domain. For example, this process is demonstrated



KVL and KCL subgraphs model the behavioral interaction of circuit modules.

Figure 1. B-graph for A_{v_d} and Z_{in} of a differential amplifier circuit.

schematically to define three linguistic values of Low, Medium and High for closed interval [0,VCC] in Figure 2.

There is one interesting thing worth mentioning here. In qualitative arithmetic, the outcome of some arithmetic operations may be "ambiguous". For example, the result of adding two "Medium" values in Figure 2 may be "Medium" or "High". The ambiguities are used here to obtain alternative options for the computation results.

Final qualitative values for specifications and quantities of each constructed circuit topology can be set to initial numerical points. Then, optimum values of the circuit parameters can be obtained by a numerical optimizer. This stage of design is not the subject of this article.

For a unique circuit representation, the basic circuit type U_t with structure $\bar{S}$, causal behavior $\bar{B}$ and the corresponding constraint set $\bar{C}$ is defined. U_t is represented in a hierarchical form as:

$$U_t : \begin{cases} S_1 & S_2 \ldots S_n \\ B_1 & B_2 \ldots B_n, n \geq 1 \\ C_1 & C_2 \ldots C_n \end{cases} \text{ or briefly as } U_t : \begin{cases} \bar{S} \\ \bar{B} \\ \bar{C} \end{cases},$$

where S_i, B_i and $C_i (1 \leq i \leq n)$ denote circuit information at level i. As a definition, B_{i+1} is an instant from B_i and S_{i+1} is an instant from S_i. As indicated in the above definition, causal relations and the associated constraints as well as the circuit structure are hierarchical. In this way, the hierarchical approach has been extended from structure to behavior. For example, Figure 3 depicts two levels of hierarchical representation of the structure and the behavior of a push-pull output stage.

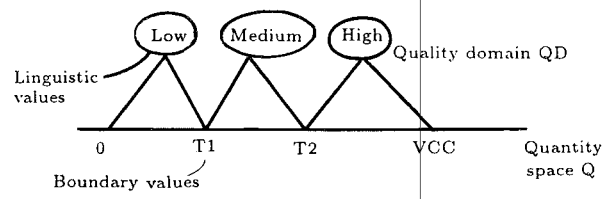


Figure 2. Mapping of quantity space Q to the quality domain.

Figure 4 illustrates the basic behavioral based design procedure within the BSDM system. With reference to this figure, a brief overview of the method and purpose of each system module is now provided [12].

Basic Behavioral Modeling

To design the desired circuit, the basic behavioral features of the circuit are first derived from the given function and specifications. The desired basic behavior is represented by a B-graph and the corresponding set of constraints.

Primary Circuit Generation

The primary circuit generator selects primary circuits or elements from the library which best match the desired behavior. Then, this module, qualitatively, evaluates the behavior of the primary circuit.

Circuit Behavior Modification

If a failure occurs, this module will be activated. It expands the initial B-graph such that all desired causal relations and/or constraints can be met. B-graph expansion provides a model which expands circuit equations into more detailed equations in order to satisfy the desired constraints. In this way, B-graph expansion provides a modified version of the initial behavior in which the initial synthesis problem is decomposed into two or more synthesis subproblems. The novel algorithms developed for B-graph expansion operate

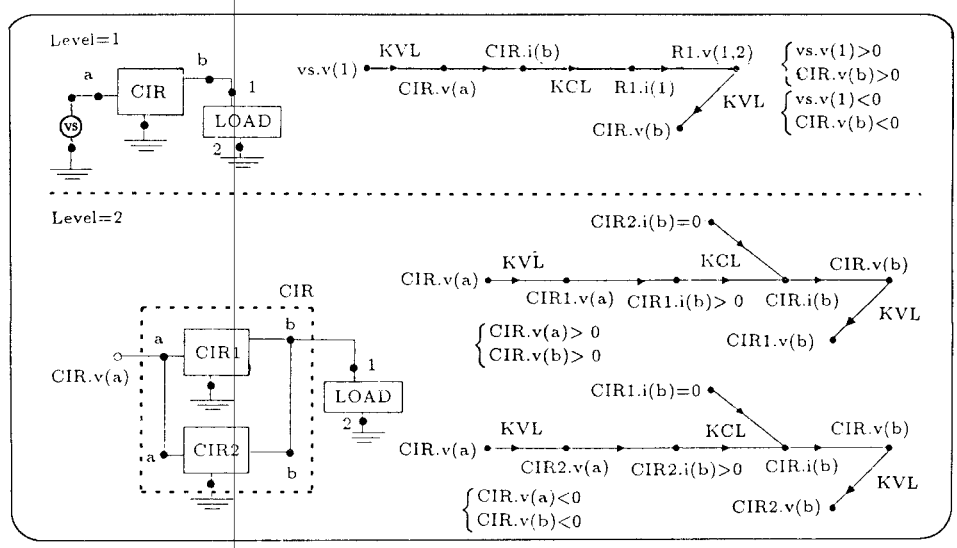


Figure 3. Hierarchical representation of push-pull output stage structure and behavior.

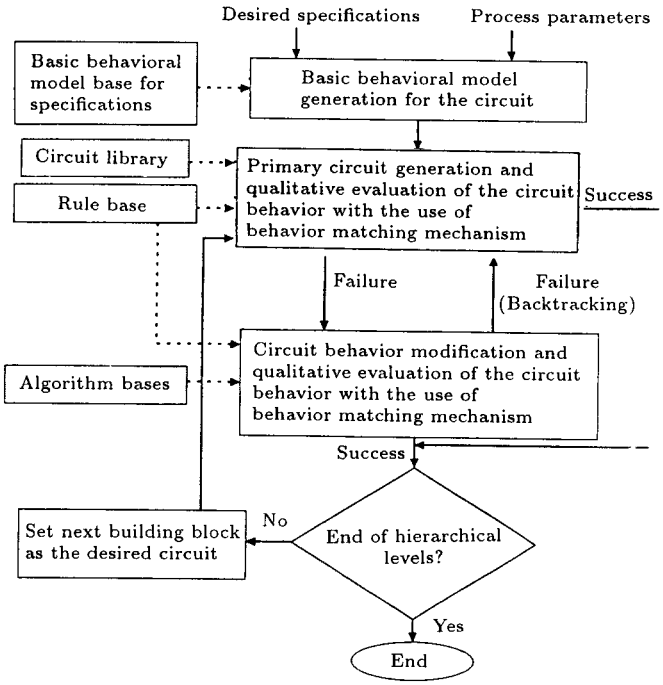


Figure 4. Design procedure within BSDM system.

such that KVL/KCL are satisfied in the resulting B-graph(s). These algorithms modify circuit structure along with B-graph expansion by decomposing a circuit module into a number of circuit modules by the use of reasoning and then interconnecting these modules in a way that is consistent with KVL and KCL. Consequently, this method is able to produce different circuit topologies without the use of a library. This is a new and important feature of the presented design theory.

The newly constructed block diagram, along with the desired behavior at a determined level, is again given to the primary circuit generator. Design procedure continues up to the complete design of the circuit.

UNIFORM MODELING OF CIRCUITS, KNOWLEDGE AND SYNTHESIS PROBLEM

Two categories of domain knowledge are modeled and used to apply heuristics as follows:

1. Rules:
   If $\bar{S}1$ and/or $\bar{B}1$ and/or $\bar{C}1$ then $\bar{S}2$ and/or $\bar{B}2$ and/or $\bar{C}2$.
   These rules give information about $\bar{S}$ and/or $\bar{B}$ and/or $\bar{C}$ according to the known properties of $\bar{S}$ and/or $\bar{B}$ and/or $\bar{C}$. KVL and KCL and heuristic rules for constraint assignment are of this nature.
2. Elements and circuits library.

This category includes electrical elements or already known circuit blocks such as a MOS transistor or a cascode amplifier stage for which U_t is known. The library in the presented method helps to increase the speed of convergence in the design process without sacrificing the robustness of the method.

Problem representation is depicted in a format similar to that of knowledge representation of the second type above. $\bar{S}$, $\bar{B}$, and $\bar{C}$ are only defined at the first level of hierarchy for a synthesis problem defined by a U_t. Design process is indeed the deriving information for the next levels.

APPLICATION OF KNOWLEDGE BASED ON CONCEPT BEHAVIOR MATCHING

To apply inference rules, inference engine of the design system must be able to determine when two circuits are the same or match [14]. A matching mechanism performs this task. This mechanism is an algorithm, which compares the information of circuits and calculates required replacement of variables to decide whether a circuit can be replaced by another circuit or not and a rule can be activated or not. At the basic behavior level, the matching mechanism is classified into structural matching, which is familiar and behavioral matching, which includes B-graph and constraint matching. If H_p is the desired behavior,

$$(H_p = \begin{cases} B_p \\ C_p \end{cases}),$$

in a synthesis problem and H_k is a known behavior,

$$(H_k = \begin{cases} B_k \\ C_k \end{cases}),$$

behavior matching evaluates H_k and checks the satisfaction of the H_p requirements.

B-graph matching compares the causal relations of two different circuit modules and applies a unification algorithm for determining the variable substitutions of B_p and B_k. These substitutions generate the maximum number of desired causal relations in B_p, each of which will be provided by a set of causal relations in B_k. Successful B-graph matching is followed by constraint matching. Constraint matching itself includes comparison, intersection and propagation of constraints on the B-graph through the circuit hierarchy [12]. Constraint propagation will be also applied once each function block is designed. This procedure determines:

- What constraints are imposed by each circuit module onto other modules (especially to those blocks that have not undergone the design process yet).
- Whether interaction of two or more designed circuit modules produces conflict.

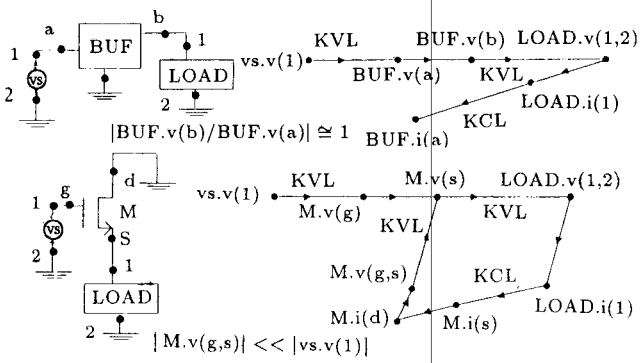


Figure 5a. Desired behavior of a voltage buffer and behavior of a CD stage.

Quantity Matching		Causal Relation Matching	
BUF	**M**	**BUF**	**M**
BUF.v(a)	M.v(g)	BUF.v(a) to BUF.v(b)	M.v(g) to M.v(s)
BUF.v(b)	M.v(s)		

$|M.v(g,s)| << |vs.v(1)| \Rightarrow M.v(s) \cong M.v(g)$

Figure 5b. Behavior matching results of U_p and U_k in Figure 5a.

- Whether the desired and imposed constraints for the behavior of a circuit module produce conflict.

As an example of behavior matching, the desired behavior of a voltage buffer and the behavior of a common drain stage and matching results are shown in Figures 5a and 5b, respectively.

B-graph matching specifies how a desired relation from an independent variable to a dependent variable in the desired circuit will be generated by a set of variables and their relations existing in the network equations of a specified circuit. The situation under which B-graph of the specified circuit does not produce some of the desired causal relations or constraints of the behavior do not satisfy some of the desired constraints, is named partial matching of B-graphs or constraints, respectively. In the simple case, the circuit modules can be replaced with other alternative modules. This is a generally used strategy for circuit modification. In the next section, a new method for circuit behavior modification based on basic behavioral model is presented.

MODIFICATION OF CIRCUIT BEHAVIOR WITH THE USE OF B-GRAPH EXPANSION

Circuit topology modification in traditional methods is limited to selecting a new topology from the library and replacing the failed configuration with the selected topology. To overcome this limitation, a new concept is defined and used, i.e., formal modification of behavior, the basis of which is to expand circuit equations into more detailed equations.

For the case of partial matching of B-graphs, it is necessary to add new subgraphs to the original B-graph (B_1) and form a new B-graph (B_2) such that complete matching occurs. Mapping this procedure from B-graph domain to the network equations is equivalent to expanding the initial desired circuit equations by adding new variables and new causal relations.

For the case of partial matching of constraints, consider a primary component with B-graph B_k and constraints set C_k. Suppose B_k is matched with the desired B-graph, but C_k embodies a subset of desired constraints set C_p. Thus, by propagating C_p and C_k through B_k a set of conflicts will remain. This means overconstrained network equations. If propagation of both C_p and C_k is done over an increased number of variables and relations, in other words, on the expanded circuit equations, conflicts can be resolved. Since nodes of B-graph denote circuit variables and branches of B-graph variable relations, adding suitable nodes and branches to B_k (i.e., expanding original B-graph) results in the expansion of circuit equations.

In general, for the case of partial matching of constraints, there will be some subgraphs with inconsistent constraints resulted from the constraint propagation mechanism. If a failed subgraph is "KVL" or "KCL", the situation implies that the interaction of two or more related circuit modules must be modified. If the failed subgraph relates to the behavior of a circuit module, the situation means that the module equations must be modified.

As the behavioral model is hierarchical, B-graph modification can be applied at each level of hierarchy. The lower the level, the more local modifications and the upper the level, the more global modifications.

B-graph is expanded with the use of novel developed algorithms. To map the B-graph expansion procedure from the B-graph domain to the structure domain, it is necessary to create new modules and interconnections whose B-graph corresponds to the added subgraphs of the initial B-graph. The resulting B-graph is a lower level instant from the desired B-graph and resulting structure is a lower level instant from the desired structure. Modification of the circuit structure in parallel to B-graph expansion has been predicted in the developed algorithms. This can be briefly summarized as follows.

For each new node during B-graph expansion, a new circuit block is generated. At the beginning, these blocks do not have any electrical connection with each other or with existing blocks. To create the appropriate connections, consistency of the resulted structure with new causal relations must be secured. To provide this consistency, structural properties of KVL and KCL are used in a formal manner. This usage is performed

in such a way that specific electrical connections are created between circuit modules for each new or expanded KVL/ KCL subgraph generated during B-graph expansion. As a result, the separate existing blocks are connected to each other so that they satisfy structural conditions of "KVL" and "KCL" subgraphs while providing new causal relations. Thus, innovative modification of the structure will be followed. B-graph expansion is realized by the use of different algorithms for each case of failure.

To illustrate the procedure, a sample algorithm which removes constraint failure will be provided here along with an example.

B-Graph Expansion Algorithm B for Constraint(s) Failure in "KVL" or "KCL"

Consider failed "KVL" and "KCL" subgraphs. For each one of these subgraphs such as B_1:

1. Create a new auxiliary source node n'_s and connect it to the sink node of the subgraph. Assume one new circuit block for n'_s.
2. Select a cause (source node) for n'_s and create a minimal path from this cause node to n'_s, or set n'_s as an independent node, i.e., without any cause. In the former case, for each "KVL" or "KCL" subgraph in the path create the corresponding interconnections between circuit modules.
3. Apply constraint propagation mechanism to the B-graph.

As an example, consider design of a DC voltage level shift with negative value and zero input and output current in MOS technology. The B-graph of voltage level shift is simply a node without any branch and the associated constraint is the qualitative value of the voltage shift. It is assumed that a NMOS transistor M, which must be operated in the saturation region, is selected as the primary component. Thus, the initial B-graph is as shown in Figure 6a. The matching mechanism locates failures in the "KCL" subgraph in which the output current is generated and in the node representing M.i(d). This is because M is not connected to any circuit blocks, thus transistor M has zero terminal currents. To modify the circuit, algorithm **B** is applied.

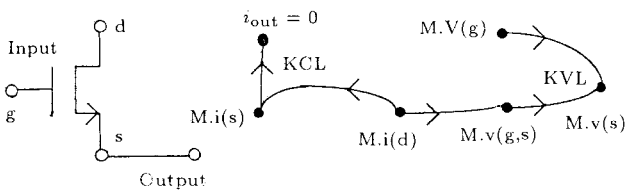


Figure 6a. Primary component for voltage level shift.

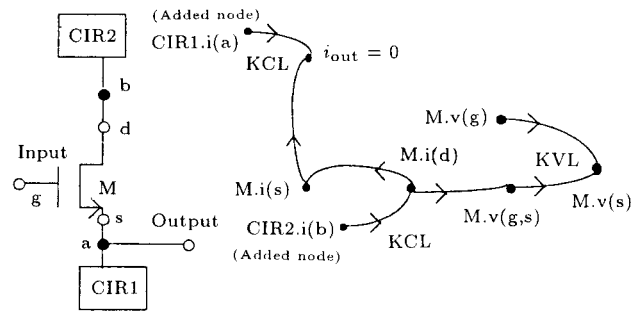


Figure 6b. B-graph expansion for initial circuit in Figure 6a leads to new circuit blocks CIR1 and CIR2.

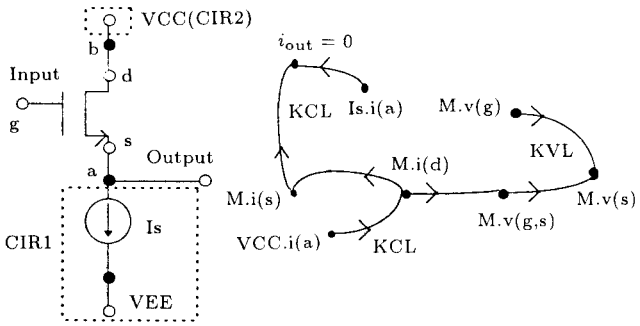


Figure 6c. Modified topology after structure generation for CIR1 and CIR2.

Application of Algorithm B

Consider the "KCL" subgraph in which "i_{out}" and the node representing M.i(d) are generated (Figure 6a).

1. Create a new auxiliary source node for node "i_{out}" and assign this new source node to CIR1.i(a). Also create a new auxiliary source node for node "M.i(d)" and assign this new source node to CIR2.i(b). CIR1 and CIR2 are new circuit blocks (Figure 6b).
2. Set CIR1.i(a) and CIR2.i(b) as independent nodes.
3. Apply constraint propagation mechanism to the B-graph.

Design of CIR1 and CIR2 blocks are new synthesis problems which are again to be solved in the same manner. An option for topology of the whole circuit is shown in Figure 6c.

SYSTEM ARCHITECTURE

In order to realize the basic behavioral based design procedure (see Figure 4), a global system architecture is chosen for BSDM as shown in Figure 7. The main system components are the user interface, knowledge and algorithm bases, S unit and M unit. S unit

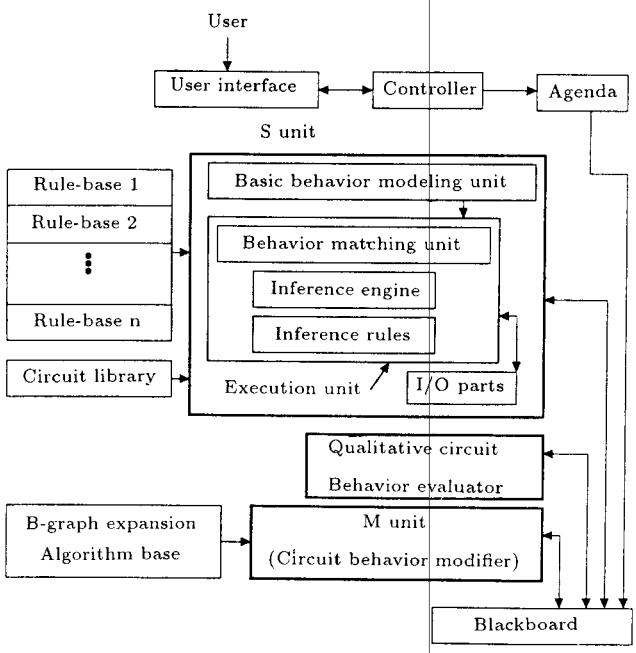


Figure 7. BSDM design system architecture.

realizes the primary circuit generation phase and M unit realizes the circuit behavior modification phase of design procedure. S unit is knowledge-based whereas M unit is essentially algorithm based.

Communication among all system components is established by the aid of a blackboard [15]. BSDM is also hierarchical, i.e., the main tasks of the design procedure are extended to a set of subtasks to be performed at S or M unit. BSDM has been realized on a PC-Pentium with the use of Borland C++ version 4.5 under Windows 95.

DESIGN PROCESS AND EXAMPLES

To obtain a perspective of design procedure, a description of relevant steps in the design is given here [16].

Design Process

Synthesis problem: $U_p : \begin{cases} S_1 & S_2 = ? \ldots S_n = ? \\ B_1 & B_2 = ? \ldots B_n = ?, n \geq 2 \\ C_1 & C_2 = ? \ldots C_n = ? \end{cases}$

Step 1: select a candidate module

$$U_k : \begin{cases} S'_1 & S'_2 \ldots S'_l \\ B'_1 & B'_2 \ldots B'_l, 1 \geq 1 \\ C'_1 & C'_2 \ldots C'_l \end{cases}$$

from library, such that $C_1 \cap C'_1 = \text{Max}$.

Step 2: Compare causality of behavior of U_p and U_k (B-graph matching).

Step 3: Examine B-graph matching.

- Case 1: $B'_1 \cap B_1 = \phi$ (complete mismatch). Backtrack.
- Case 2: $B'_1 \cap B_1 = B_1$ (problem is solved from causal relations point of view). Go to Step 4.
- Case 3: $B'_1 \cap B_1 = B'', B'' \neq B_1$ (partial matching). Apply appropriate B-graph expansion algorithm.
  Result: $B_1 = B_{2_1} \cup B_{2_2}$, $S_1 = S_{2_1} \cup S_{2_2}$, $C_1 = C_{2_1} \cap C_{2_2}$ or $\bar{B} = \bar{B}1 \cup \bar{B}2$, $\bar{S} = \bar{S}1 \cup \bar{S}2$, $\bar{C} = \bar{C}1 \cap \bar{C}2$.
  New synthesis problems are:

$$U_{p1} : \begin{cases} \bar{S}1 \\ \bar{B}1 \\ \bar{C}1 \end{cases} \text{ and } U_{p2} : \begin{cases} \bar{S}2 \\ \bar{B}2 \\ \bar{C}2 \end{cases} \text{ .Go to Step 1.}$$

Step 4: Compare constraint sets C and C' (constraint matching).

- Case 1: $C_1 \cap C'_1 \neq \phi$ for each specification. (Synthesis problem has been solved.) Stop.
- Case 2: $C_1 \cap C'_1 = \phi$ for at least one specification. Apply appropriate B-graph expansion algorithm (given as following).
  Result: $B_1 = B_{2_1} \cup B_{2_2} \ldots \cup B_{2_m}, S_1 = S_{2_1} \cup S_{2_2} \ldots \cup S_{2_m}, C_1 = C_{2_1} \cap C_{2_2} \ldots \cap C_{2_m}$ or $\bar{B} = \bar{B}1 \cup \bar{B}2 \ldots \cup \bar{B}m, \bar{S} = \bar{S}1 \cup \bar{S}2 \ldots \cup \bar{S}m, \bar{C} = \bar{C}1 \cap \bar{C}2 \ldots \cap \bar{C}m$.
  New synthesis problems are:

$$U_{p1} : \begin{cases} \bar{S}1 \\ \bar{B}1 \\ \bar{c}1 \end{cases}, U_{p2} : \begin{cases} \bar{S}2 \\ \bar{B}2, \ \ldots, \\ \bar{C}2 \end{cases}$$

$$U_{pm} : \begin{cases} \bar{S}m \\ \bar{B}m, \text{ Go to Step 1} \\ \bar{C}m \end{cases}$$

Here algorithm **A** for expanding B-graph in the case of failure in constraints is presented. Then, two design examples executed in BSDM environment with the use of algorithms **A** and **B** are given. Details of algorithm **A** are provided for the second example.

Algorithm A for Constraints Failure

This Algorithm pinpoints the cause(s) of constraint failure(s) on the B-graph and removes the cause(s) as follows:

1. Select a subgraph B_f related to a circuit equation in which a failure has occurred and has sink node n_k and all the related source nodes and branches.
2. For one or more source nodes in the subgraph :
   - Disconnect an incident branch to n_k from a source node n_s.

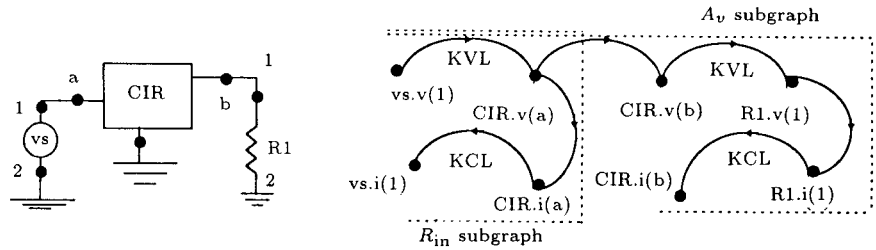


Figure 8a. Design problem definition (AC design).

- Create a new source node n'_s for n_k and connect it to n_k by the use of a new branch. Assume one new circuit block for n'_s.
- Select a cause (source) node for n'_s and create a minimal path from this source node to n'_s. For each "KVL" or "KCL" subgraph in the path create the corresponding interconnections between circuit modules.

3. Apply constraint propagation mechanism to the B-graph.

If considered failed subgraph is "KVL" or "KCL", the algorithm **B** is also applicable.

Example 1

A basic design example is shown in Figure 8. The problem is the design of a buffer stage with an $A_v \cong 1$, high R_{in} and low R_l (Low load). Figure 8a displays the problem statement and Figure 8b shows the small library content. Figure 8c represents the selected candidate module from the library. This module fails to meet the desired R_{in} if bias current is chosen high and fails to meet the desired A_v if bias current is chosen low. The library includes only three basic configurations for one-stage amplifiers. Here, it is shown how algorithm **B** generates a group of new configurations which satisfy the desired specifications.

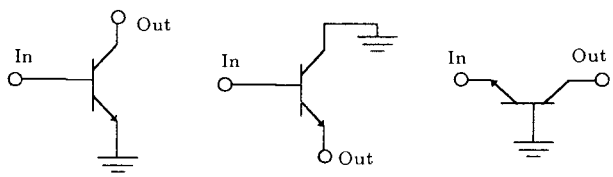


Figure 8b. Library content (only one stage topologies).

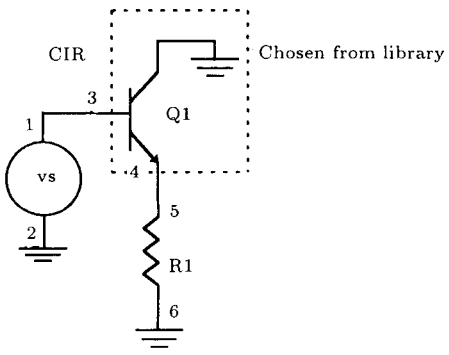


Figure 8c. Selected candidate module from library.

Application of Algorithm B for Failure in R_{in}

1. With respect to the "KCL" subgraph of R_{in} in Figure 8a,
   - Create a new auxiliary source node for the node representing "vs.i(1)" and assign this new source node to CIR2.i(5). CIR2 is a new circuit block.
   - Select CIR1.v(4) as a cause for CIR2.i(5) and create a path from the node representing "CIR1.v(4)" to the node representing "CIR2.i(5)" (Figure 8d).
2. Apply constraint propagation mechanism to the expanded B-graph.

Continuing the design procedure for CIR1 and CIR2, with respect to the given library, will result in the circuit of Figure 8e as a solution to the given problem.

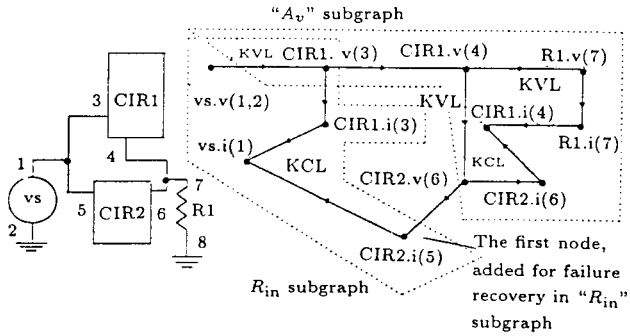


Figure 8d. Expanded B-graph starting from "R_{in}" subgraph and corresponding new inferred schematic.

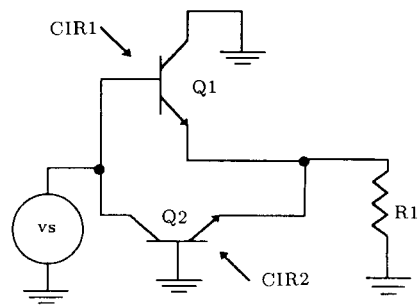


Figure 8e. One option of possible circuit structures for increasing R_{in} resulted from block schematic in Figure 8d.

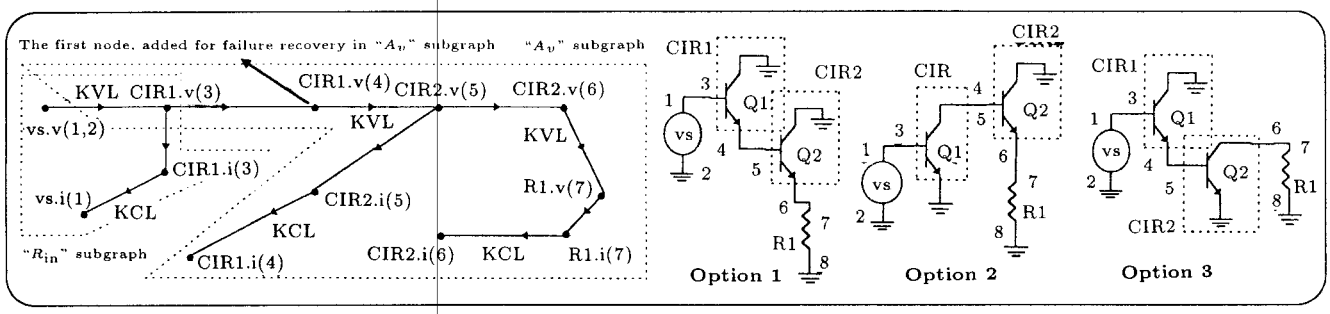


Figure 8f. Expanded B-graph starting from "A_v" subgraph and three options of new constructed structures.

The circuits of Figure 8f could also be derived as other alternative solutions, if A_v subgraph (instead of R_{in} subgraph) was chosen as the cause of failure.

Example 2

The design of a low offset differential amplifier is given in Figures 9 to 13. The desired qualitative specifications are: A_{vd} = High, R_{in} = Medium, P.D. = Low, input offset voltage = Low, $|\Delta I_{0\,\max}|$ = High, $|\Delta V_{0\,\max}|$ = High, input DC voltage = 0. Figure 9 depicts the component library, which only includes basic modules and electrical elements. In this example, the following design steps are followed.

1. The design problem model is generated.
2. The library is searched for an initial topology whose range of specifications maximally match the constraints of the desired specifications.
3. For each failed specification or quantity, the corresponding subgraph is determined and expanded by algorithm **A**. As a result, a new B-graph, associated constraints and associated block diagram, including interconnection of circuit blocks AMP1 and AMP2, are innovatively generated in a formal manner. AMP1 will correspond to the selected topology in Step 1. For this module, corresponding behavior and structure will be placed at the next level of circuit hierarchy.
4. Constraints of the desired specifications and constraints of the behavior of resolved module AMP1 will be set for the circuit B-graph. The total set of constraints are then propagated through the complete B-graph of the total circuit. Also, heuristic rules of the domain knowledge are applied to derive further constraints for the behavior of AMP1 and AMP2. For low offset amplifier heuristic sets, the offset voltage of the first stage is to be low and the absolute value of gain of AMP1 is to be large enough, i.e., greater than or equal to a medium value. As a result of the constraint propagation, the behavior interaction of AMP1 and AMP2 modules through the associated "KVL" and "KCL" subgraphs will impose a set of new constraints on AMP2.
5. Again design Steps 2 to 4 are repeated to design and innovatively modify function block AMP2.
6. Design of each circuit block is a new synthesis problem which is solved in a similar way. This process continues until all circuit blocks at the lower levels of hierarchy are designed.

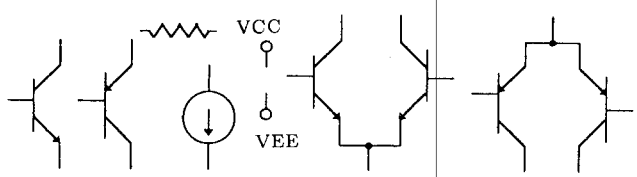
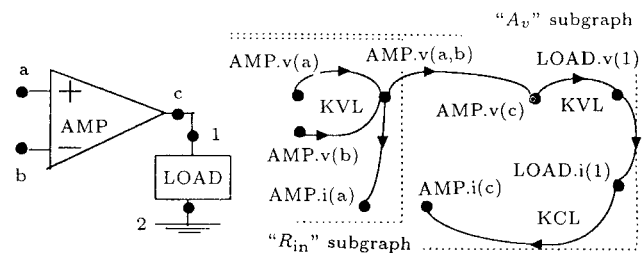


Figure 9. Library content used for design.

Figure 10a demonstrates the design problem model. There are two options for the primary solution, npn differential pair and pnp differential pair, both of which fail to meet the desired A_{vd}, $I_{0\,\max}$ and $V_{0\,\max}$; npn differential pair is selected with qualitative specifications under the following certain loading conditions. $A_{vd} = -M$ with Load $\geq$ Medium, R_{in} = Medium, P.D. $\leq$ Medium, $I(c)_{dc}$ > Very Low for transistors, $|\Delta I_{0\,\max}|$ = Medium and $|\Delta V_{0\,\max}|$ = Medium. Furthermore, transistors of a differential pair must operate in active region. Then, A_v subgraph of the circuit is expanded. This subgraph itself includes some subgraphs. If a "KVL" or "KCL" subgraph is selected, interaction of the primary circuit with the LOAD or voltage source module will be corrected. If input voltage-output



Desired specifications: A_{vd}=High, R_{in}=Medium, P.D.=Low, Input offset voltage=Low, $|\Delta I_{0\,\max}|$=High, $|\Delta V_{0\,\max}|$=High, Input DC voltage = 0.

Figure 10a. Design problem definition.

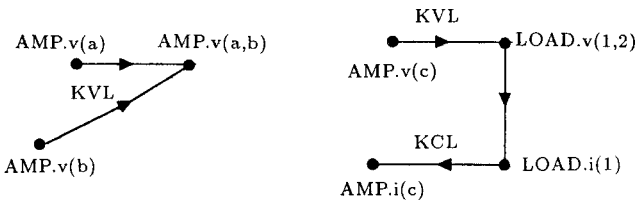


Figure 10b. Disconnecting incident branch to "AMP.v(c)" from "AMP.v(a,b)".

voltage subgraph is selected, primary circuit behavior will be corrected. However, in each case, initial circuit block will be expanded to a topology of interconnected function blocks. Here, input voltage-output voltage subgraph is expanded. Figures 10b to 10h illustrate the details of algorithm **A** to show how the circuit B-graph is expanded at the first level, through which a two-stage amplifier is intelligently generated. This algorithm is much more complex than the simple process of connecting digital circuit building blocks. This is due to the fact that the analog circuit behavior is much more complex compared to the behavior of digital circuits.

Application of Algorithm A for Failure in A_{vd}

With respect to the input voltage-output voltage subgraph of A_{vd} in Figure 10a, and for the source node representing "AMP.v(a,b)" (n_s):

- Disconnect incident branch to the node representing "AMP.v(c)" (n_k) from n_s (Figure 10b).
- Create a new source node n'_s for n_k and connect it to n_k by the use of a new branch. Assign AMP2.v(e,f) to n'_s. AMP2 is a new circuit block at the second level of AMP hierarchy (Figure 10c). Select AMP.v(a,b) as a cause for AMP2.v(e,f) and create a path from the node representing "AMP.v(a,b)" to the node representing "AMP2.v (e,f)". For each "KVL" or "KCL" subgraph in the path create the corresponding interconnections between circuit modules. Here, the most simple connections on the B-graph are constructed using three branches as following (Figure 10d):
  - First branch from AMP.v(a,b) to AMP1.v(a,b), which is assigned to KVL. AMP1 is a new circuit block at the second level of AMP hierarchy.
  - Second branch from AMP1.v(a,b) to AMP1.v (c,d), which is assigned to input voltage-output voltage relation of AMP1 behavior.
  - Third branch from AMP1.v(c,d) to AMP2.v(e,f), which is assigned to KVL and denotes an interaction between AMP1 and AMP2.
- Create KCL subgraphs corresponding to "KVL" subgraphs B1 and B2 in Figure 10d (Figure 10e). "KCL" subgraph B'2 denotes another interaction between AMP1 and AMP2.
- Select causes for new nodes "AMP1.i(a)" and "AMP2.i(e)". Here a straightforward case is selected as following (Figure 10f):

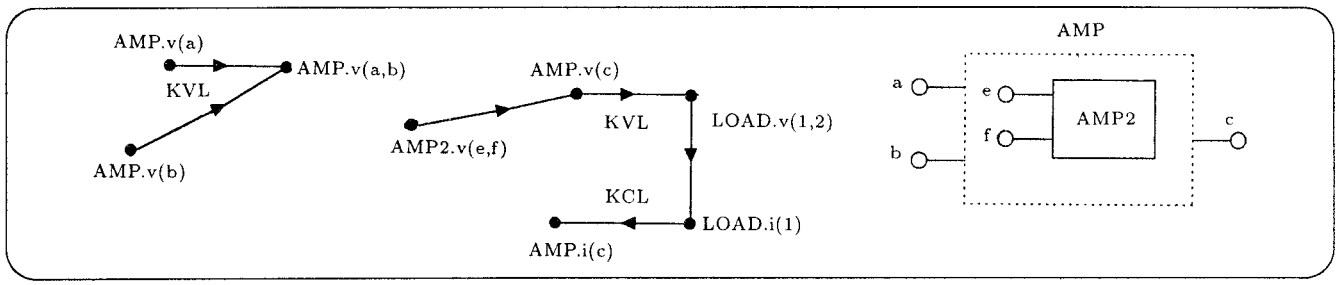


Figure 10c. Creating new circuit block AMP2 and assuming "AMP2.v(e,f)" as a cause for "AMP.v(c)".

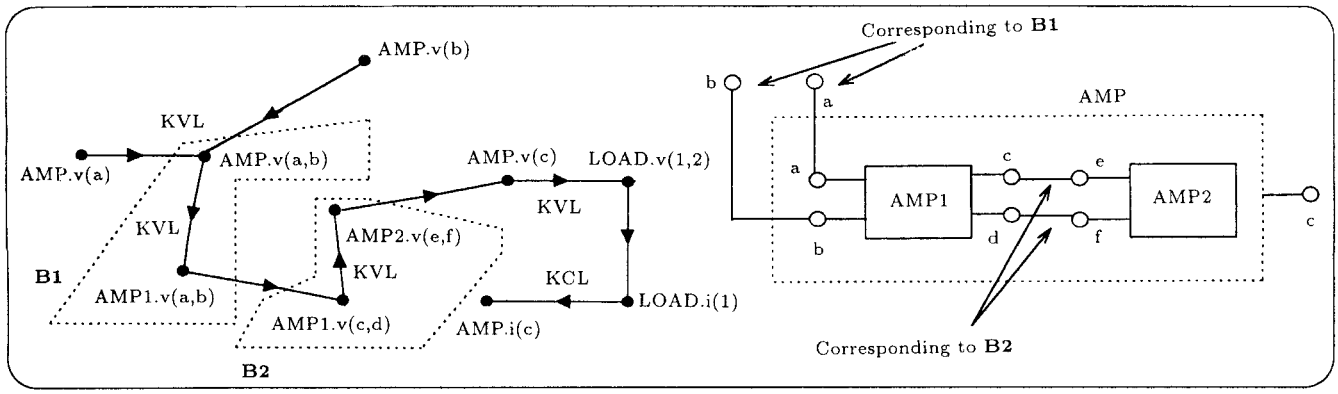


Figure 10d. Creating the most simple connections from "AMP.v(a,b)" to "AMP2.v(e,f)" on B-graph and corresponding new electrical interconnections.

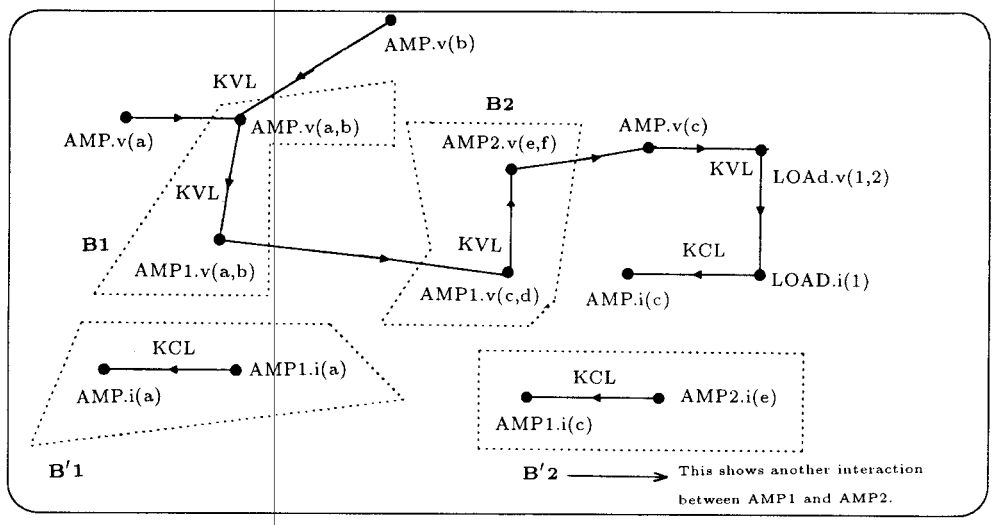


Figure 10e. Creating "KCL" subgraphs corresponding to "B1" and "B2" in Figure 10d.

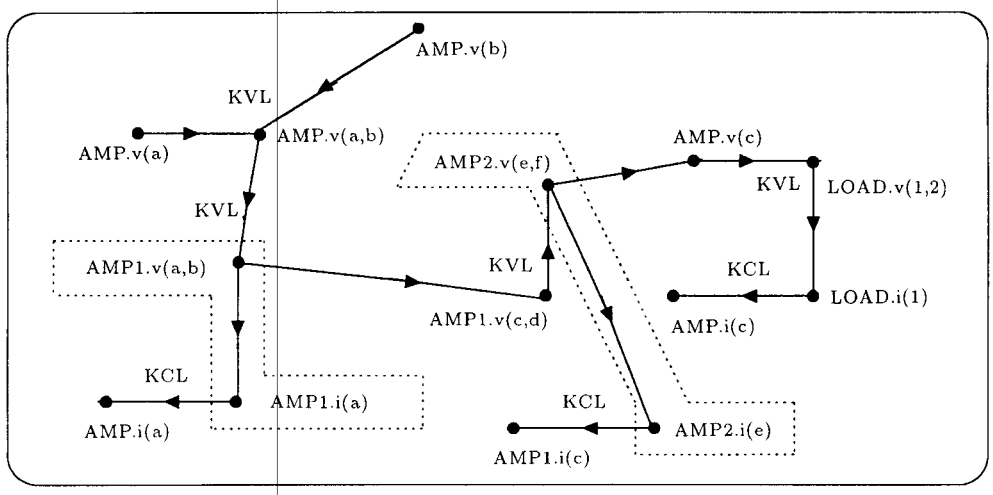


Figure 10f. Selecting causes for new nodes "AMP1.i(a)" and "AMP2.i(e)".

- AMP1.v(a,b) as a cause for AMP1.i(a), which leads to R_{in} subgraph for AMP1.
- AMP2.v(e,f) as a cause for AMP2.i(e), which leads to R_{in} subgraph for AMP2.

• Replace causal relation from AMP2.v(e,f) to AMP.v(c) with the most simplest possible path. Here the simplest path is constructed from two branches as following (Figure 10g):

- First branch from AMP2.v(e,f) to AMP2.v(g). "g" is a new terminal assigned to module AMP2. This branch is assigned to input voltage-output voltage relation of AMP2 behavior.
- Second branch from AMP2.v(g) to AMP.v(c), which is assigned to "KVL" subgraph.

Constraint propagation is done for the circuit B-graph of Figure 10g, which is shown in Figure 10h. Imposed constraints on AMP2 resulted from AMP2 interaction with AMP1 are also illustrated.

To design AMP2, design steps are again repeated for AMP2 module. Here, there is only one option for the primary circuit of AMP2, that is pnp differential pair. This is because of the required DC levels for input and output terminal voltages of AMP2. These constraints are imposed by AMP1 module and small signal voltage gain of AMP2. As DC power dissipation in AMP1 can be set "Low", there is no failure in power dissipation, the same thing is also applied for AMP2. PNP differential pair fails to meet the desired $I_{0\,\max}$ and $V_{0\,\max}$. Again algorithm **A** is applied to expand AMP2 B-graph through which an amplifier stage AMP22 and an output buffer stage BUF are intelligently generated. "KVL" subgraph is selected to modify interaction of AMP2 and LOAD module. Figure 10i depicts the complete circuit B-graph and the associated block diagram at the second level of circuit hierarchy. At the last phase of the algorithm, the total set of constraints through the expanded B-graph

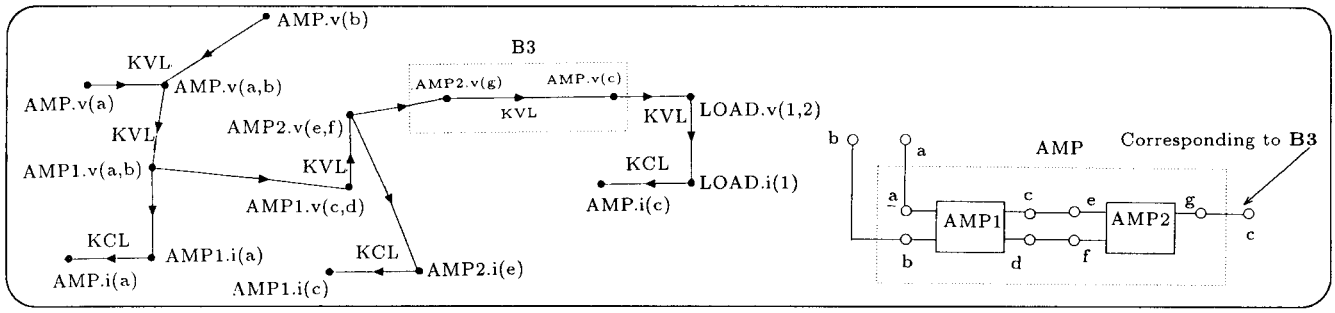


Figure 10g. Creating the most simple connections from "AMP2.v(e,f)" to "AMP.v(c)" on the B-graph and corresponding new electrical interconnections.

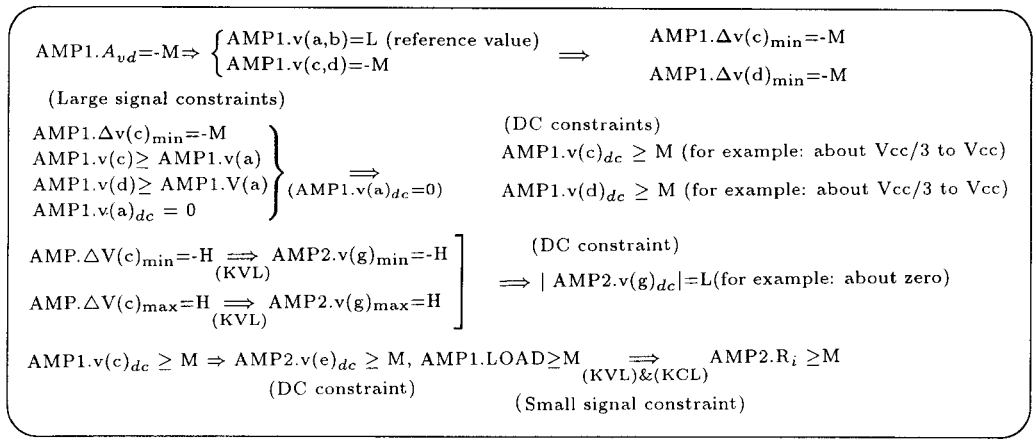


Figure 10h. Constraint propagation for the B-graph of Figure 10g.

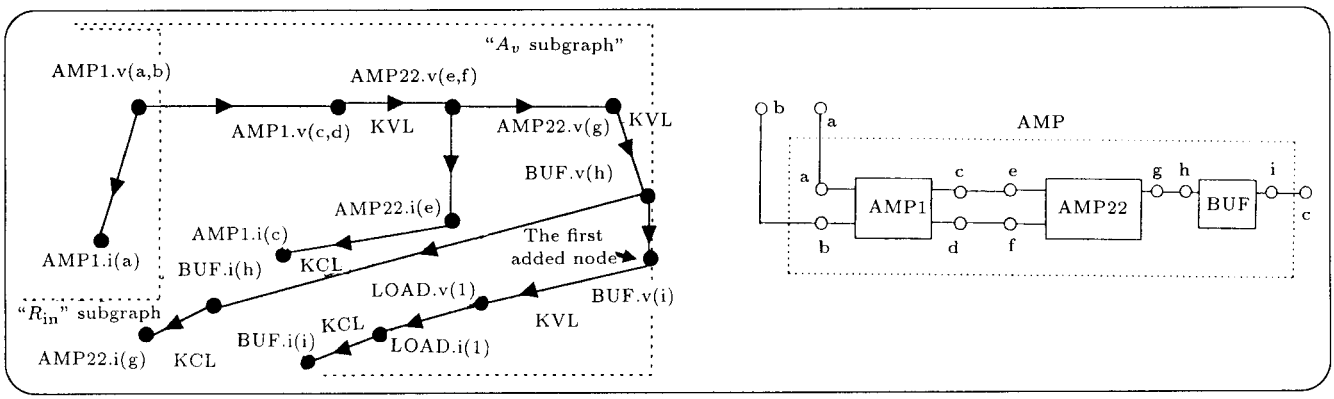


Figure 10i. Final circuit B-graph and structure resulted from consequent execution of expansion algorithm **A**.

and then the complete B-graph of the total circuit are propagated. As DC power dissipation in AMP2 can be set "Low", there is no failure in power dissipation and power dissipation for BUF is also set "Low", i.e., bias current of subcircuit BUF is Low.

Application of Algorithm A for Failure in $I_{0\,max}$ and $V_{0\,max}$

With respect to the input voltage-output voltage subgraph of A_v in Figure 10g, which has a sink node representing "LOAD.v(1,2)" and for the source node representing "AMP2.v(g)" (n_s):

- Disconnect incident branch to the node representing "LOAD.v(1,2)" (n_k) from n_s.
- Create a new source node n_s' for n_k and connect it to n_k using a new branch. Assign BUF.v(i) to n_s'. BUF is a new circuit block.
- Select AMP2.v(g) as a cause for BUF.v(i) and create a path from the node representing "AMP2.v(g)" to

the node representing "BUF.v(i)". For each "KVL" or "KCL" subgraph in the path create the corresponding interconnections between circuit modules (Figure 10i).

Figure 11 shows the generated lower level blocks design for the first stage of amplifier guided by domain knowledge (two options of possible solutions). Gradual generation of lower level blocks is obtained using expansion algorithms **A** and **B** for both small and large signal behavior of the circuit as was described for AMP module. Using resistor and transistor loads are both for removing inconsistency in the "KCL" subgraph for large signal collector current of transistors in differential pair. Finally, Figure 12 represents two alternatives of the resulted topologies for the amplifying stages.

Other options for circuit topologies are also obtainable from different linguistic values for the quantities. As illustrated in Figure 10h, two linguistic values are possible for R_{in} of AMP2. If "Medium" R_{in} is considered for AMP2, a simple pnp differential pair with Medium R_{in} can be selected for it. Topologies in Figure 12 are based on Medium R_{in} for AMP22 resulted from Medium R_{in} for AMP2 . If "High" R_{in} is considered for AMP2, after selecting the pnp differential pair primary topology for AMP22, constraint propagation leads to failure in "R_{in}" and, as a result, inconsistent constraints for the "R_{in}" subgraph of AMP22. Again algorithm **A** or **B** must be applied to modify circuit behavior and structure. One of the solutions is obtained in the form of inserting a buffer stage as shown in Figure 13 along with the associated expanded B-graph.

CONCLUSIONS

A new method for automatic analog circuit design based on behavioral modeling and associated design environment (BSDM) was presented. Behavioral modeling is conducted with the use of a new graph, called B-graph. B-graph and associated qualitative constraints model different aspects of analog circuit behavior. The presented method is capable of locating the causes of failure on the B-graph and then algorithmically modifying the B-graph to achieve behavior modification. B-graph expansion algorithms simultaneously modify the circuit behavior and structure without the use of library, which is a novel feature of the method. Mixed behavioral and structural modification makes the method innovative in the design process with the capability of producing alternatives for the desired circuit structure. Heuristic rules are also applied to

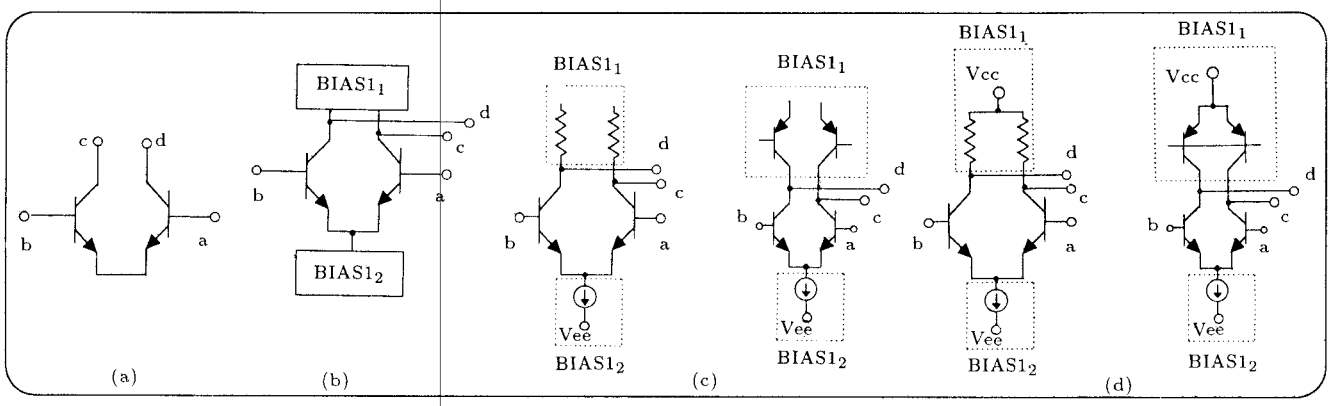


Figure 11. Internal circuitry of AMP1 resulted from expansion algorithm **B** for failure removal of "KCL" subgraphs in DC B-graph (two options are shown in (c) and (d).

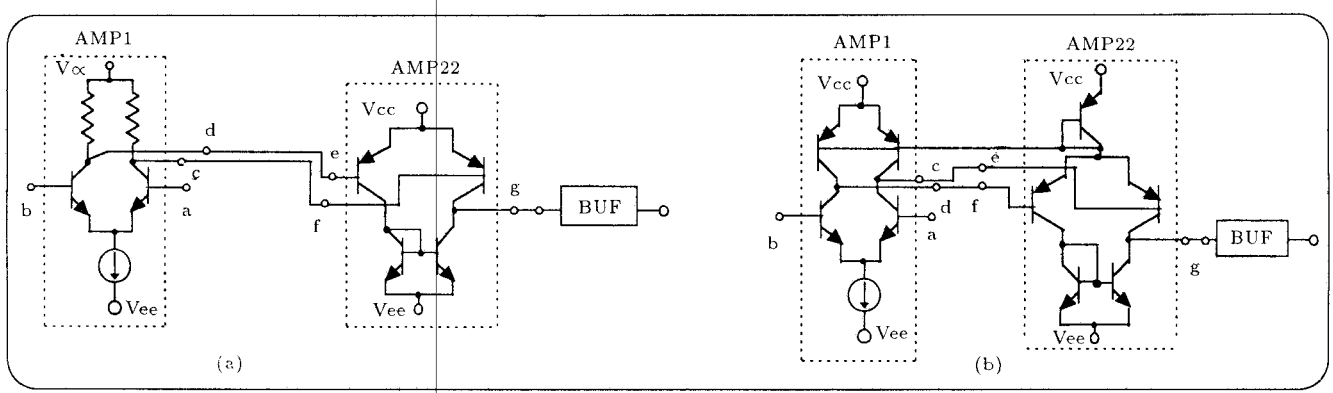


Figure 12. Two options for final inferred topology of amplifying stages for the desired circuit AMP.

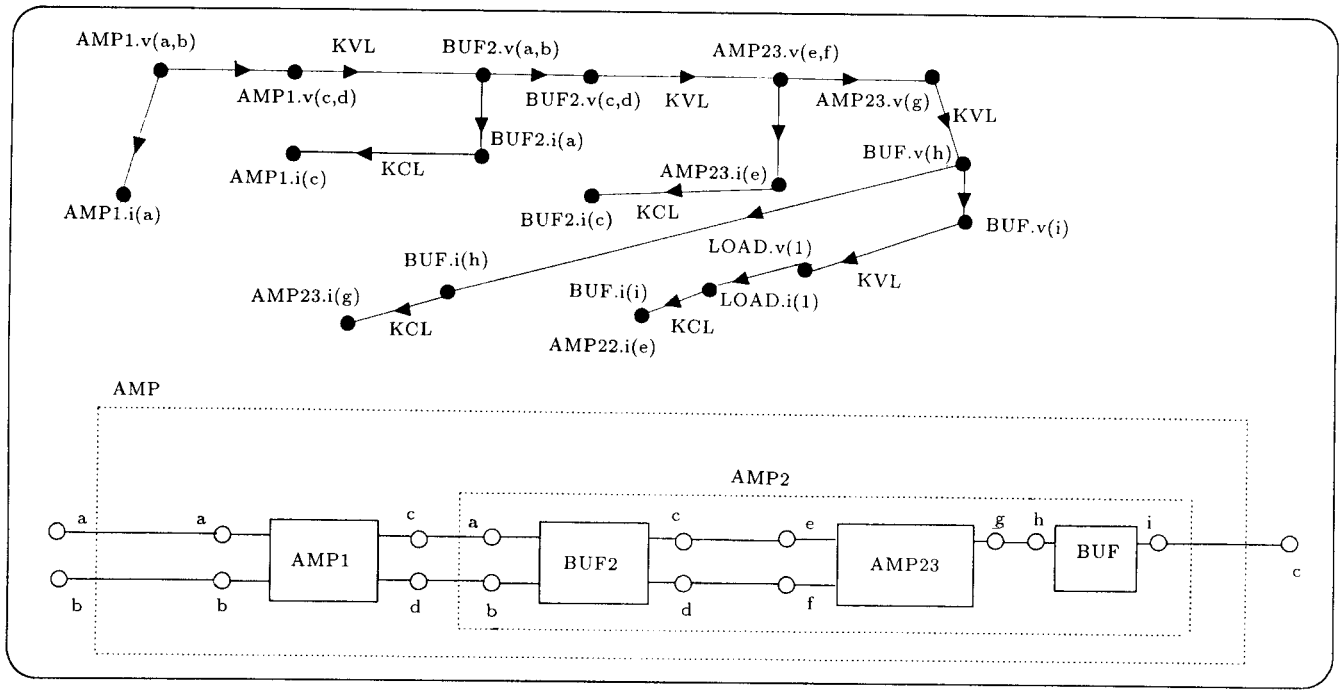


Figure 13. Another option for circuit B-graph and structure, resulted from considering high R_{in} for AMP2.

the design process. This makes the method both algorithmic and knowledge-based. For future work, frequency information must also be included. For this purpose, it is necessary to add another B-graph to the circuit behavioral model.

REFERENCES

1. Toumazou, C. and Makris, C.A. "Analog IC design automation, Part I and Part II", *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems*, **14**(2), pp 218-254 (1995).
2. Klinke, R., Fiedler, H.L., Senior, A. et al. "Rule-based analog circuit design", *IEEE Design Automation European Conference* (1992).
3. Degrave, M.G.R. et al. "IDAC: an interactive design tool for analog CMOS circuits", *IEEE J. of Solid State Circuits*, **SC-22**(6), pp 1106-1115 (1987).
4. Fatehy, El. Turkey and Perry, E.E. "BLADES: an artificial intelligence approach to analog circuit design", *IEEE Trans. on Computer Aided Design*, **9**(2), pp 680-692 (1989).
5. Harjani, R., Rutenbar, R.A. and Carley, L.R. "OASYS: a framework for analog circuit synthesis", *IEEE Trans. on Computer Aided Design*, **8**(12), pp 1247-1265 (1989).
6. Swings, K., Dnnay, S. and Sansen, W. "HECTOR: a hierarchical topology construction program for analog circuit synthesis", *IEEE 1991 Custom Integrated Circuits Conference* (1991).
7. Wittmann, R., Schardein, W., Senior, A. et al. "Application independent hierarchical synthesis methodology for analog circuits", *European Design Automation Conference* (1994).
8. Horta, N.C. and Franca, J.E. "A methodology for automatic generation of data conversion topologies from algorithms", *IEEE International Symposium on Circuits and Systems* (1994).
9. Franca, N.E., Lanca, M.A. and Franca, J.E. "Op-Cadsys: an open tool for automatic synthesis of circuit components for data converters", *IEEE 37th Midwest Symposium on Circuits and Systems* (1994).
10. Milzner, K. "An analog circuit design environment based on cooperating blackboard systems", *J. of Applied Intelligence*, **1**, pp 179-194 (1991).
11. Williams, B.C., *Qualitative Reasoning About Physical Systems*, D.G., Bobrow, Ed., Cambridge, MA, MIT Press, pp 281-346 (1985).
12. Shojaei, M. and Sharif-Bakhtiar, M. "Automatic design of analog circuits based on a behavioral model", *IEEE 1998 Canadian Conference on Electrical and Computer Engineering*, Waterloo, Ontario, Canada (May, 1998).
13. Williams, B.C. "A theory of interactions: unifying qualitative and quantitative algebraic reasoning", *J. of Artificial Intelligencce*, **51**, pp 39-94 (1991).
14. Rich, E. and Knight, K., *Artificial Intelligence*, Second Edition, McGraw-Hill (1991).
15. Jackson, P., *Introduction to Expert Systems*, Second Edition, Addison-Wesley (1990).
16. Shojaei, M. and Sharif-Bakhtiar, M. "A method for automatic design of analog circuits based on a behavioral model", *IEEE International Symposium on Circuits and Systems*, Monterey, California, USA (May, 1998).