KNOWLEDGE BASED EXTENSION OF DIADES SYSTEM FOR THE ANALYSIS AND SYNTHESIS OF TGC CIRCUITS

Edmund Pierzchala *, *Wojciech Nikiel* **, *Marek A. Perkowski* *

* Portland State University, Portland, Oregon
P.O. Box 751, 97 207, tel. (503) 725-386

** Warsaw Technical University
Nowowiejska 15/19, Warsaw, Poland

ABSTRACT

The paper presents a new and uniform approach to the computer-aided analysis and synthesis of analog circuits. The system is specialized for Transconductance-Grounded Capacitance Circuits (TGC). The analysis is a transformation from a Signal-Flow Graph (SFG) to a Transfer Function (TF). It is based on a step-by-step transformation from SFG to TF by reductions of nodes: summation, multiplication, and feedback. Symbol manipulation of multi-variate rational functions is used. The synthesis is a process of transforming a *LC*-ladder filter SFG to a TGC circuit. Heuristic synthesis procedures, inverse to the ones used in the analysis, search the solution space of equivalent SFG's and are theoretically able to find the optimal solution. The synthesis method includes three stages: *SFG labeling*, *synthesis of the SFG branch transfer functions*, and *SFG repolarization*. The application of some synthesis rules is illustrated on examples.

1. INTRODUCTION

In this paper, an integrated (or uniform) approach to the analysis and synthesis of analog, continuous-time domain filters is proposed. Transconductance-Grounded Capacitor circuits technique [25-28] is chosen as an implementation basis, because it reveals many features desirable for the automated synthesis. It is particularly well suited to the standard CMOS technology, what makes it a natural choice for mixed (analog-digital) circuits realizations. On the other hand, it is not limited to this technology and can be implemented in BiCMOS and bipolar technologies as well. All the circuits in TGC are built in two different basic building blocks: transconductance element (transconductor), which realizes Voltage Controlled Current Source, and grounded capacitor. The variety of high-frequency analog and mixed circuits can be built with the help of the TGC technology in a systematic way.

With the exception of the CAD systems for the Switched Capacitor circuits [11], the existing Design Automation systems for analog circuits are not tightly dedicated to one particular circuit technology. Therefore, although these systems are more general, they are not robust enough to be used for the synthesis of more complicated, real-life circuits.

The proposed in this paper extension to the DIADES design automation system [16,17] (that has been used for many years used as a tool for synthesizing digital circuits) will cover symbolic analysis and synthesis of TGC analog circuits (analog filters) and mixed circuits in the future.

2. SYSTEM OUTLINE

Phases of the analysis and synthesis processes are shown in Fig. 1. *Heuristic symbolic manipulation procedures* perform the desired transfer function transformations: factorization, continuous fraction expansion, etc.

Innovative trait of the proposed methodology is the fact, that the synthesis process on each stage is actually a reverse of the analysis process. The most difficult part of the synthesis process is a transformation of a transfer function (TF) to signal-flow graph (SFG). This is because the synthesis of SFG from given TF, although can be presented as a process reverse to the analysis, requires decisions about SFG topology, that is not described by TF. Therefore, the application of a particular *mapping rule* from TF to SFG requires *heuristics*, guiding the process of mapping rules selection. Decisions taken on behalf of the heuristics are based on the desired characteristics of the created SFG, like sensitivity to elements variations, and on the circuit complexity (number of the elements used).

Since the symbolic analysis of TFs and SFGs is not dependent on the method of later implementation of the SFG in particular circuit technology, large part of the system is independent of the used circuit realization techniques. Mapping of SFG to TGC also doesn't force the choice of any particular technology of TGC circuits realization.

The *mapping rules* describe the relationship between elementary SFG connections and corresponding to them TFs. Four mapping rules are shown in Fig. 2a-d. The fourth one is actually a special case of the feedback rule (Fig. 2c) when $\frac{1}{T_1} \rightarrow 0$. It is the only rule that can be applied when a TF with the numerator of the higher order than the denominator has to be realized out of elementary circuits that realize only TFs with the numerator of the order not higher than that of the denominator.

In this paper there is a convention accepted, that current signal is represented in the SFG as ●, and voltage signal as ○.

The most powerful rule is a feedback rule (Fig. 2c). It is particularly convenient for the realization of the continuous fraction expansion of a TF; with components realizing only integrating TFs (i.e. of the form $1/sC$), TF with both integration and differentiation in the denominator can be realized.

The summing rule (Fig. 2a) is useful whenever addition operation, in many cases handled by the feedback rule, is realized directly.

The multiplying rule (Fig. 2b) is used to invert the sign of TF, for example. It is also a powerful synthesis tool when the realized TF is in the factorized form.

SFG synthesis in TGC technology (Transconductors and Grounded Capacitors - [25-28]) is based on the TGC realization of elementary SFG arms. Fig. 3 shows some examples of TGC configurations (building blocks) that can be regarded as elementary. The very basic set of TGC building blocks consists of only two elements: transconductor (Voltage Controlled Current Source) and grounded capacitor. They will be called *basic elementary TGC building blocks*. SFG's branches realized by them are shown in Fig. 3a. Each TF can be transformed to the SFG with all the arms directly realizable by the basic elementary TGC building blocks. It is convenient, however, to augment the set of elementary building blocks by some combinations of the basic elements. They are shown in Fig. 3b.

3. ANALYSIS

The analysis system (SFG analysis) has been written in the programming language Lisp, since it requires symbol and graph manipulation and arbitrary precision arithmetic. It transforms the SFG (created automatically or by the user), back to a single reduced TF.

The presented algorithm is quite efficient since it is not based on matrix calculus (the matrices are very sparse) but on the step-by-step, knowledge-controlled local SFG transformations.

There are three types of rules. The lowest level rules operate on polynomials and rational functions (addition, multiplication, factorization, division, etc). The next level operates on symbolic transmittances (summing rule, multiplying rule, feedback rule).

The highest level includes heuristic rules that control the selection of the second level rules in order to generate the final transmittance in the simpliest form. Those rules are based on the graph structure and correspond to basically three methods:

- Variable Reduction (substitution of a variable being the left side of a symbolic equation by the right part of this equation in all other equations, thus reducing the number of equations by one).
- Equation Solving (solving symbolically a single equation of the form $x = ax + b$, where x is a symbolic variable and a, b are, most generally, arbitrary multi-level symbolic transitional formulas of many variables, usually sums of multi-level rational functions).
- Formula Simplification of a transitional symbolic formula.

The structure of the SFG decides which of the three methods is used where and when. In the analysis process the number of the nodes is gradually reduced by means of the step-by-step application of those methods, until an SFG with only one node is created, which is described by resultant TF. At the same time the expressions corresponding to the nodes are simplified. The method was also used to systems with many inputs and outputs.

The Simplification algorithm was written as a modification of a program to find regular expressions from a Regular Grammar (a Finite State Machine) since a one-to-one correspondence among those two problems exists [18]. We used, therefore, the methods to locally optimize regular expressions, which lead to the simplification of the transitional and final transfer functions. Simplification uses also a classical algebraic simplification algorithm, which applies polynomial factorization and division as well.

CH2819-1/90/0000-0989$01.00©1991 IEEE

One of the problems solved by the system in the symbolic analysis phase is the reduction of rational functions. For one-variable polynomials Collins created several efficient algorithms based on quite sophisticated mathematical theories. The newest of the Collins' algorithms uses holomorphisms, divider theory and interpolation to simplify polynomials of many variables [4]. In order to achieve sufficient speed of rational function manipulation we use both mathematical techniques and efficient symbol manipulation programming. Therefore, two subsystems of DIADES, ABRA and SOPS, deal with rational functions. ABRA has been designed especially to perform calculations on polynomials and rational functions of many variables. (ABRA was used for both Laplace and Z-transform to design systems of automatic control, linear automata, filters, and generation of cyclic codes). It uses efficient mathematical algorithms of a narrow scope. In turn, SOPS is a general-purpose symbol manipulation system, similar to MACSYMA [14] but simpler. It performs expression differentiation, substitution, simplification, general-purpose pattern-matching formula transformations, operations on symbolic matrices, and many other operations, not specialized for rational functions. To achieve high efficiency, SOPS does not perform reduction of rational functions, which is in turn a task ABRA was constructed for. Many problems are then solved by the cooperation of these two systems and it is a task of the programmer to assign judiciously where and how the operations are executed. SOPS was used to symbolically differentiate and simplify various classes of expressions, including symbolic analysis of sensitivity, pattern-matching substitutions and tensor calculations [12,13].

4. SYNTHESIS

In the scheme proposed in this paper there is a general feature of SFG synthesis on the basis of given TF. In the first stage of the system development, described in this paper, we assume that this feature is used only for the realization of non-elementary SFG branches, but the SFG itself is given as the input data. This is because there exist excellent SFG topologies, like for example the topology resulting from the top-level SFG simulation of *LC*-ladder filters (see [25-28]). In the future development of the system the SFG creation will be handled to the full extent.

4.1. SIGNAL-FLOW GRAPH IMPLEMENTATION IN TGC TECHNOLOGY

We will assume the following:

- the Signal-Flow Graph (SFG) corresponding to the *LC*-ladder prototype of the filter is given;
- the only elements used for the TGC filter implementation are the transconductor and the grounded capacitor;
- transmittance of each ladder arm (described by a transfer function) can be realized in an arbitrary topology; our goal is to find the one with the minimal number of elements.

The procedure will be explained on the example of the filter presented in [25-27].

Given is an *LC*-ladder prototype of the filter, shown in the Fig. 4a (the ladder arm transmittances are given further in this paper). The corresponding SFG is shown in Fig. 4b.

The SFG synthesis process can be partitioned into three phases. In the first phase, called *SFG labeling*, the physical realization of the signal represented by each SFG node is decided and each node is then labeled by the current or voltage. The second phase is the *synthesis of the SFG branch transfer functions*. The third phase is called *SFG repolarization* and is aimed at removing redundant signal inversions from the SFG.

4.1.1. SFG LABELING

The labeling is based on the physical realizability of two different elementary nodes. An elementary node is a node with only one incoming signal and an arbitrary number of outgoing signals, or with a one outgoing signal and an arbitrary number of incoming signals (Fig. 5a). Node of the first type will be realized as a voltage node i.e. there will be a circuit node, voltage of which, measured to the ground, will represent the signal of the SFG node. Node of the second type will be realized as a current node and will correspond to a wire in the electrical circuit, which conducts current representing the signal associated with the SFG node. This is the only way how the elementary nodes can be implemented in a physical circuit. If a given node has both more than one incoming signal and more than one outgoing signal, it must be split into two elementary nodes, as shown in Fig. 5b. Finally, the input and output nodes of the SFG must be labeled as voltage nodes (unless stated otherwise).

Because of the above rules, the nodes in the SFGs of the mapping rules are also labeled by current and voltage labels. The only unlabeled mapping rule is the multiplying one (Fig. 2b). Labeling for this rule should be chosen in the way that is most convenient to realize both branch transmittances.

Additionally all adjacent current nodes connected by SFG branches labeled by *1* are contracted into one current node at this stage. This operation is called *node contraction*.

Since the SFG given in in the example has only the elementary nodes, the labeling can be done immediately. Fig. 6 shows the labeled SFG, after contraction of the nodes I_1 and D.

4.1.2. SYNTHESIS OF THE SFG BRANCH TRANSFER FUNCTIONS

Second step is the realization of the branch transmittances. For this purpose, the mapping rules are applied to the transfer function of the ladder arm.

In the presented example, let us first implement the impedance Z_1. This will help understand the aforementioned possibility of applying more than one mapping rule to given transfer function.

$$Z_1 = \frac{1}{sC_4 + \frac{sC_3}{s^2L_3C_3+1}} \tag{1}$$

The above function could be realized as a serial connection of transfer functions of lower orders, obtained by factorization. In this example, however, we will realize it as a SFG with a feedback loop, as shown in Fig. 7a. It turns out, that the feedback arm can be implemented in the best way as the feedback connection, (Fig. 7b). Two possible implementations of the inner feedback loop, leading to different realizations of the whole transfer function Z_1, are shown in Fig. 7c. They realize the inner transmittance (between nodes D and E) with opposite algebraic signs. These implementations are equivalent with the respect to the number of elements. It turns out, however, that choosing the second one (which corresponds to the negated transfer function of the inner feedback loop) leads to a whole feedback loop with two negative transconductance elements, instead of negative and positive ones. Since negative transconductance element is cheaper (it contains one instead of three transconductors), we choose the SFG shown in Fig. 7d. The corresponding TGC circuitry is shown in Fig. 7e.

This illustrates one possible local transformation of the SFG, basically related to the nodes polarization. At this point we will notice, that realizing Z_1 is cheaper by two transconductance elements than realizing $-Z_1$. This information will be used later, in the node repolarization phase.

To realize the transfer function Z_2, given below (see [26,27]), we proceed as follows.

$$Z_2 = \frac{1}{sC_8 + \frac{1}{R_L} + \frac{1}{sL_7}} \tag{2}$$

As before, we use the feedback mapping rule. The result is shown in Fig. 8. It follows, that the realization of $-Z_2$ (actually shown in Fig. 8b) is by two transconductance elements cheaper than the realization of Z_2.

Transfer function Y_1, given below, can be realized in a similar way as Z_2.

$$Y_1 = \frac{1}{sL_1 + R_s + \frac{1}{sC_2}} \tag{3}$$

The result is shown in Fig. 9. The only difference between Z_2 and Y_1 is that the former one is a voltage-to-voltage transfer function, while the latter is a current-to-current one. Finally we notice, that it is cheaper to realize $-Y_1$ than Y_1.

The best implementation of the Y_2, given below

$$Y_2 = \frac{1}{\frac{1}{sC_6} + \frac{sL_5}{s^2L_5C_5+1}} \tag{4}$$

is by application of the "inverting" mapping rule (Fig. 2d). This is necessary, since the order of the numerator is higher than the order of the denominator, and none of the other mapping rules (in their general form) can lead to the SFG realizable in TGC technology. So at first we realize the reciprocal of the Y_2, as a voltage-to-current transfer function, to be able to obtain the desired current-to-voltage one. The realization is based on the summing rule (Fig. 2a), and the feedback rule (Fig. 2c). The result is shown in Fig. 10. Realization of $-Y_2$ is cheaper than that of Y_2.

4.1.3. SFG REPOLARIZATION

At this stage we will "repolarize" branches of the SFG to be able to apply minimal realizations of branch transmittances (i.e. either Z or –Z) in as many branches as possible. We know the desired polarity of vertical ladder arms, and the desired polarity of the horizontal ones is –1 for the voltage-to-current arms. Table 1 shows the actual and desired polarities of the SFG arms.

Table 1

branch	is	desired	branch	is	desired	branch	is	desired
V_iA	+	-	V_2A	-	-	V_2B	+	-
V_oB	-	-	AD	+	-	I_2D	-	-
I_2V_o	+	-	DV_2	+	+	BI_2	+	-

There are four simple rules for local repolarizations in the SFG, which are shown in Fig. 11. We will start the repolarization phase from node I_2. Changing the polarity of I_2 signal allows us to change both BI_2 and I_2V_o polarities, which is desired. No change is propagated to the nodes B and V_0. Now we have either to change the polarity of I_2D branch (which is not desired), or to change the polarity of the D node. Changing the polarity of the node D we change the polarity of A and V_2. The last change leads to the desired change of the polarity of the V_iA and V_2B, both of them desired. The resulting SFG is shown in Fig. 12.

In the presented case repolarization was extremely easy due to the primary SFG polarities assignment. In general case it is not possible to get all the desired polarity changes. In such a case, the number of undesired polarities should be minimized.

Final realization of the filter is shown in Fig. 13. It contains 2 transconductance elements less than the realization presented in [26,27]. Basic topology of the resulting TGC realization of the filter is the same (i.e. it is a simulation of the ladder topology of the filter), so that the filter retains its desirable sensitivity features, inherited from the *LC*-ladder prototype.

In the above sequence of steps leading to the SFG realization in TGC technology, step 2 is apparently the best candidate for a heuristic solution.

5. HEURISTIC CONTROL MECHANISM FOR THE MAPPING RULES APPLICATION

As it was mentioned in the section 2, the mapping rules application in the SFG synthesis phase involves a decision process of appropriate rules selection. Although the exact contents of the heuristic rules has to be determined experimentally and is still in the process of development, the nature of the search process involved here is fairly well understood and an appropriate control mechanism can be proposed.

Besides the application of the mapping rules to the TF, there are symbolic algebraic transformations of the TF performed. They were not shown in the presented example, for the sake of the explanation simplicity. In the real synthesis process, however, before a matching rule can be applied, the TF expression must be transformed to the form that matches left-hand side of the rule. This is the first reason of the existence of many paths to the solution. If the TF is in the form of a sum or a product, each component of the expression can be realized in a different way, i.e. by application of different sequences of the mapping rules. These component realizations, however, are supposed to be interconnected in the circuit being synthesized, therefore their realizations are interdependent. Since there are four mapping rules that potentially can be applied at each step (maybe after reformulating the TF expression by means of algebraic transformations), one step of the TF synthesis consisting of n components (terms in a sum or a product expression) can be done in $4n$ different ways. Since the solution requires several transformation steps, the solution space of the task may be too large for the exhaustive search.

Since the relevant characteristics of the synthesized SFGs are of symbolic nature, it is difficult to propose numerical evaluation functions for the solution path selection. The control strategy proposed here is very similar to the one applied for symbolic algebraic equations solving [3].

The strategy was originally called Meta-level Inference, and is based on separation of the transformation rules (algebraic rules for TF transformations and the rules for matching TF to the SFG) into groups which realize specific kinds of synthesis operations. For instance, all the transformation rules in charge of the factorization of the TF are grouped together. Other set of rules is responsible for finding the most promising reformulation of the TF expression for the purpose of the feedback matching rule application.

On the higher level of control, called Meta-level, there is a set of heuristic rules deciding which kind of the synthesis operation should be pursued on the current step of the synthesis process, for instance, whether to try various factorization possibilities or to find the best way of the application of the feedback rule.

The Meta-level approach has proven its superiority over other methodologies for the symbolic algebraic equation solving. Because of the similar nature of the SFG synthesis process, it seems to be well suited to this task, as well.

6. REFERENCES

[1] E. Berkcan, F. Yassa, "Towards Mixed Analog/Digital Design Automation: a Review", *Proc. IEEE Symposium on Circuits and Systemss*, 1990.

[2] R. J. Bowman, "A Rule Based Approach for Adapting Analog Macrocells to Specified Circuit Contexts", *30th Midwest Symposium on Circuits and Systems*, 1988.

[3] A. Bundy, B. Welham, "Using Meta-level Inference for Selective Application of Multiple Rewrite Rule Sets in Algebraic Manipulation", Artificial Intelligence **16**, pp. 189-212, 1981.

[4] Collins, G.E., "The Calculation of Multi-variable Polynomial Resultants, *JACM.*, Vol. 18, 1971.

[5] M. Degrauwe and al., "The ADAM Analog Design Automation System", *Proc. IEEE Symposium on Circuits and Systemss*, 1990.

[6] J. E. Franca, R. Schaumann, "High Frequency Integrated Signal Processing Combining Switched Capacitor and Continuous-Time Networks", *preprint*, 1990.

[7] R. Harjani, R. A. Rutenbar, R. Carley, Analog Circuit Synthesis for Performance in OASYS, *Proc. IEEE ICCAD-88.*

[8] L. Huelsman, "Personal Computer Symbolic Analysis Programs for Undergraduate Engineering Courses", *Proc. IEEE Symposium on Circuits and Systemss*, 1989.

[9] L. P. Huelsman, P. M. Hallinan, T. Ruhe, "GASCAP - a Symbolic Circuit Analysis Program for the Personal Computer", *30th Midwest Symposium on Circuits and Systems*, 1988.

[10] J. G. Kenney, R. Carley, "CLANS: A High-Level Synthesis Tool for High Resolution Data Converters", *Proc. IEEE Symposium on Circuits and Systemss*, 1988.

[11] A. Konczykowska, M.Bon, "SCYMBAL2: a Portable Computer Program for Efficient All-symbolic Hierarchical Analysis of Large Multiphase Switched-Capacitor Networks", *Proc. of ECCTD'83*, Stuttgart.

[12] A. Krasinski, M. Perkowski: "ORTOCARTAN - A new computer program for algebraic calculations," *Computer Physics Communications* 22, pp. 269-271, 1981.

[13] A. Krasinski, M. Perkowski: "ORTOCARTAN - a new computer program for analytic calculations in general relativity," *General Relativity and Gravitation*, Vol. 13, No. 1, pp. 67-77, 1981.

[14] Lewis, E., "An Introduction to MACSYMA", *Laboratory for Computer Science*, MIT Cambridge, April 25, 1978.

[15] R. Nawrocki, "Building Set for Tunable Component Simulation Filters with Operational Transconductance Amplifiers", *30th Midwest Symposium on Circuits and Systems*, 1988.

[16] M.A. Perkowski and al., "Integration of Logic Synthesis and High-Level Synthesis", *Proc. IEEE Symposium on Circuits and Systemss*, 1989.

[17] M. A. Perkowski and al., "DIADES - a High-Level Synthesis System", *Proc. IEEE Symposium on Circuits and Systemss*, 1989.

[18] M. A. Perkowski, J.E. Brown, "Automatic Generation of Don't Cares for the Controlling Finite State Machine from the Corresponding Behavioral Description", *Proc. ISCAS'90*, New Orleans, May 1-3, pp. 1143-1146.

[19] C. S. Park, R. Schaumann, "A high-frequency CMOS linear transconductance element", *IEEE Trans. Circ. Sys.***CAS-33**, pp. 1132-1138, 1986.

[20] S. J. Seda, M. G. R. Degrauwe, W. Fichtner, "A Symbolic Analysis Tool for Analog Circuit Design Automation", *Proc. IEEE ICCAD-88.*

[21] Jyuo-Min Shyu, A. Sangiovanni-Vincentelli, "ECSTASY: A New Environment for IC Design Optimization, *Proc. IEEE ICCAD-88.*

[22] F. G. Safar, F. W. Stephenson, R.W. Steer, "A PC-Based Program for the Interactive Design of Cauer Filters", *30th Midwest Symposium on Circuits and Systems*, 1988.

[23] E. Stuffle, H. Zhang, J. Bourquin, "A Multifunctional Program for the Design of Chebyshev Filters Using a Personal Computer", *30th Midwest Symposium on Circuits and Systems*, 1988.

[24] R. E. Stuffle, "PC-SNAP: A Microcomputer Program for Symbolic Network Analysis", *30th Midwest Symposium on Circuits and Systems*, 1988.

[25] M. A. Tan, R. Schaumann, "Design of a General Biquadratic Filter Section with Only Transconductances and Grounded Capacitors", *IEEE Trans. Circ. Sys.*, vol. 35, No. 4, pp. 478-480, April 1988.

[26] M. A. Tan, *Design and Automatic Tuning of Fully Integrated Transconductance-Grounded Capacitor filters*, Ph.D. Thesis, Univ. of Minnesota, 1988.

[27] M. A. Tan, R. Schaumann, "Simulating general Parameter *LC*-Ladder Filters for Monolithic Realizations with only Transconductance Elements and Grounded Capacitors", *IEEE Trans. on Circuits and Systems*, vol. 36, no. 2, February 1989.

[28] M. A. Tan, R. Schaumann, "A Reduction in the Number of Active Components Used in Transconductance Grounded Capacitor Filters", *Proc. IEEE Symposium on Circuits and Systemss*, 1990.

[29] G. A. Veirman, R. G. Yamasaki, "Fully Integrated 5 to 15 MHz Programmable Bipolar Bessel Lowpass Filter", *Proc. IEEE Symposium on Circuits and Systemss*, 1990.

[30] C. L. Winder, R. E. Massara, "A Design Assistant Approach to the Implementation of Analogue Integrated Circuits, with Particular Reference to Switched Capacitor Filters", *30th Midwest Symposium on Circuits and Systems*, 1988.

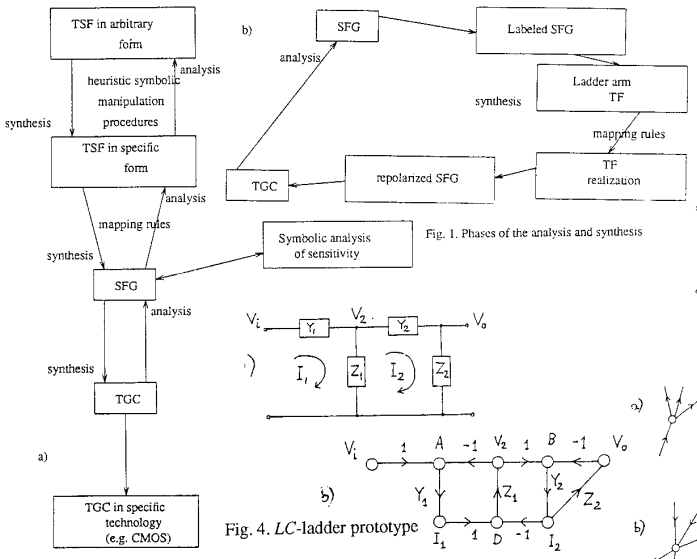

Fig. 1. Phases of the analysis and synthesis

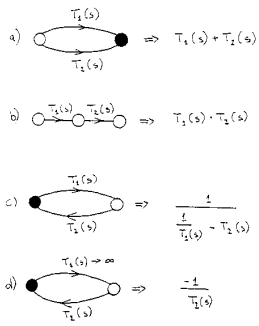

Fig. 2. Mapping rules

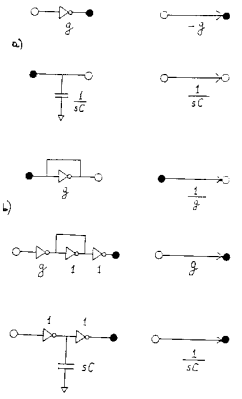

Fig. 3. TGC building blocks

Fig. 4. *LC*-ladder prototype

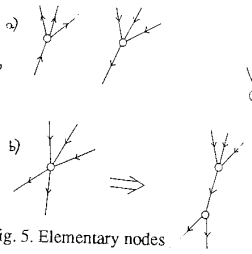

Fig. 5. Elementary nodes

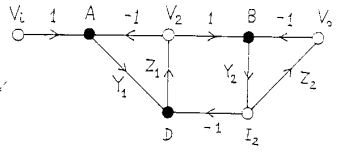

Fig. 6. Labeled SFG

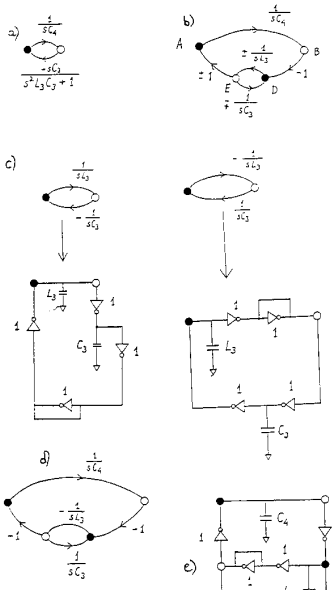

Fig. 7. Z_1 implementation

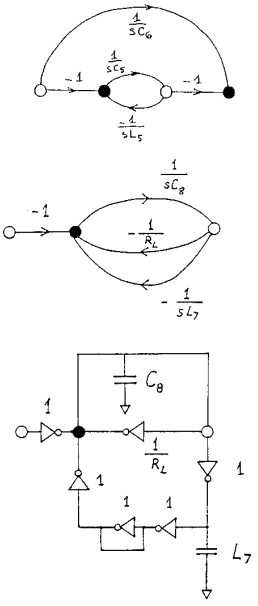

Fig. 8. Z_2 implementation

a) SFG of $\frac{1}{Y_2}$

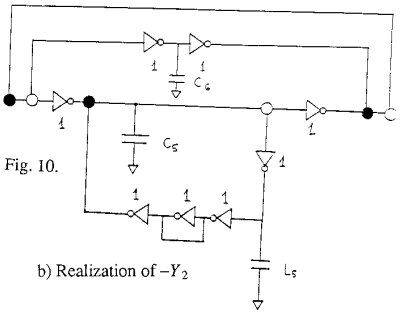

Fig. 10.

b) Realization of $-Y_2$

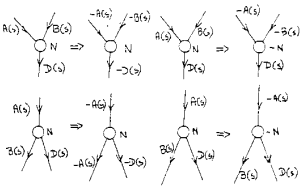

Fig. 11. Basic repolarization rules

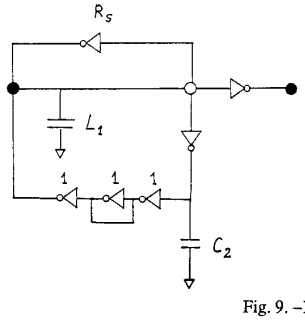

Fig. 9. $-Y_1$ implementation

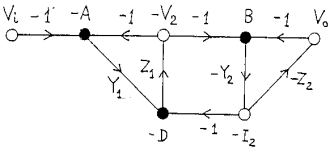

Fig. 12. Repolarized SFG

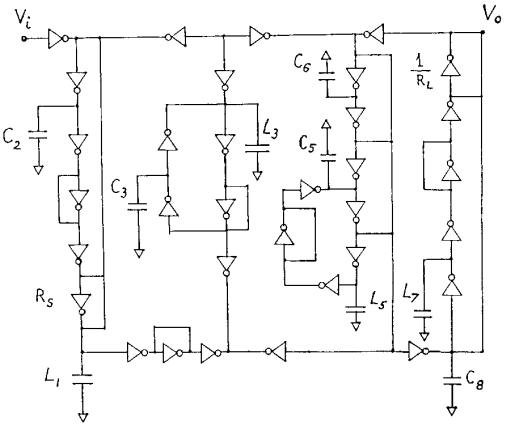

Fig. 13. The filter realization - all unlabeled transconductances have value 1