Delft University of Technology
Computational Physics Report Series

A design strategy for the synthesis of high-performance instrumentation amplifiers

J. Stoffels, C. van Reeuwijk

report number CP-96-002

ISSN 1382-5224

Published and produced by:
Computational Physics Section
Faculty of Applied Physics
Delft University of Technology
Lorentzweg 1
2628 CJ Delft
The Netherlands

Information about Computational Physics Report Series:
reports@cp.tn.tudelft.nl

Information about Computational Physics Section:
http://www.cp.tn.tudelft.nl/

A design strategy for the synthesis of high-performance instrumentation amplifiers

J. Stoffels and C. van Reeuwijk*

June 21, 1996

Abstract

In this paper, the design strategy for a circuit synthesis program is described. Unlike other synthesis programs, the program searches in an extremely large set of possible circuits (over 1,000,000 possible circuit configurations), and is not restricted to one device technology. It uses mathematically sound search rules instead of heuristic ones to ensure a high rate of success in the designs. Since it is intended for the design of high-performance amplifiers, it uses detailed models and accurate approximations of the circuit behavior. These features require considerable computational effort, and therefore it is essential to use a refined search strategy, so that the time for a design run is still acceptable. This paper describes the design and implementation of that strategy. To show the practical usefulness, a design example is given.

1 Introduction

The design of analog circuits is much more time-consuming than for digital circuits. The most important reason for this is that digital gates can be described by simple behavior equations, which simplifies manual circuit design and circuit synthesis software. As a consequence, a large range of tools have been developed for digital design, and designers of digital circuits usually can concentrate on high-level design, and do not need knowledge of the lower design levels.

Since the behavior models that are required for analog design are more complicated, the available software for analog design is much less advanced. As a consequence, analog circuit design requires knowledge of all design levels, i.e. the physical, device, circuit and layout level.

A number of attempts have been made to improve this situation by providing better software support for analog circuit design. Examples of such programs are BLADES [3], IDAC [2], OAC [8], ODIN [15], OPASYN [4], and SEAS [6]. All of these programs have at least one of the following disadvantages:

1. They only use CMOS devices.
2. They consider only a limited set of circuit configurations. For example, IDAC considers 'more than 40' configurations.
3. They use overly simplified models of the circuit behavior. Therefore, they cannot push a technology to its limits, as is necessary for more demanding applications.
4. Their design strategies are based on heuristic rules.

*When doing the research described in this paper, J. Stoffels and C. van Reeuwijk were with the University of Technology in Delft, Faculty of Electrical Engineering, P.O. Box 5031, 2600 GA Delft, The Netherlands. Research for this technical report was supported by the Dutch founding institutions FOM and STW, and by the European Community as part of the ESPRIT program.

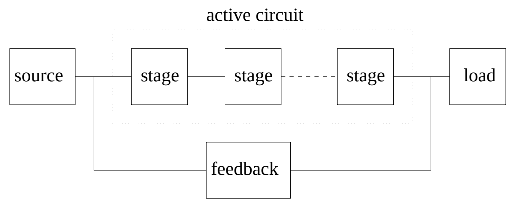

Figure 1: the general amplifier configuration used by Ampdes.

Within our research group, systematic design theory has been developed for a large part of analog circuit design, and we were convinced that it should be possible to develop a program for the design of high-performance building blocks based on this theory. To demonstrate this, we have written a prototype design system, *Ampdes*, to design high-performance small-signal amplifiers. It has the following advantages over the programs mentioned above:

- It is technology independent.
- It considers a more general class of circuits.
- It uses accurate models of the circuit behavior.
- It uses design strategies that are much less heuristic.
- Only minimal specifications are required from the user.

An outline of the design theory used in Ampdes is given in section 2. An overview of Ampdes is given in section 3.

By searching a more general class of circuits and using more accurate behavior models, Ampdes will inevitably require more computation time than other design programs. To alleviate this problem, a design strategy is used that tries to avoid detailed and time-consuming analysis steps without altering the results of the program. This design strategy is described and contrasted with that of other programs in section 5. The necessary background is given in section 4. The implementation of Ampdes is described in section 6. A design example of Ampdes is shown in section 7.

2 Outline of the design method

The design method of Ampdes is derived from the design method described in [7]. Most of the statements in this outline are justified there. See also [1].

2.1 goal

The goal of an amplifier is to convert the signal from the source to a signal that is suitable for the load. In general, both the input signal and the output signal may be represented by a number of physical quantities, for example current, voltage, charge or power. In Ampdes, however, we restrict ourselves to voltages and currents.

The conversion to be done by the amplifier may require a change of signal quantity, amplitude and spectrum. An ideal amplifier performs this conversion without loss of information, but in practical amplifiers there will always be some loss of information through noise, bandwidth limitations and distortion of the amplifier. Moreover, practical amplifiers are sensitive to parameter variations in their components. Obviously, the design strategy should minimize these disturbances.

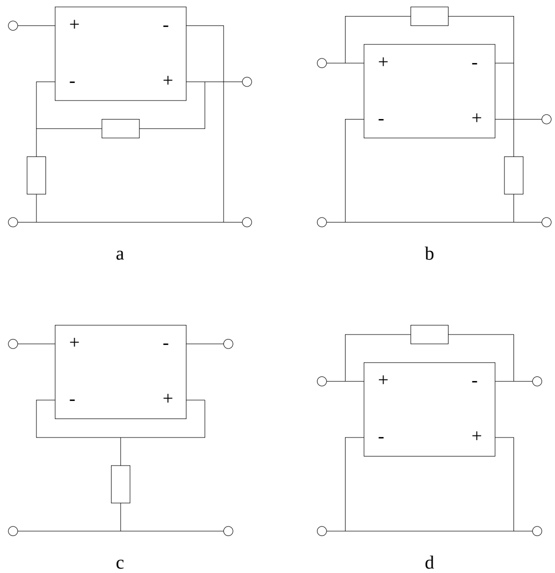

Figure 2: the possible amplifier configurations: (a) voltage amplifier, (b) current amplifier, (c) voltage-to-current amplifier, and (d) current-to-voltage amplifier.

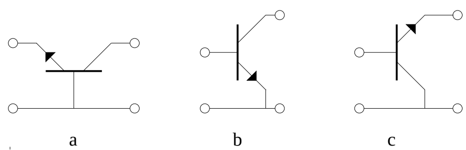

Figure 3: the possible stage configurations: (a) CB stage, (b) CE stage, (c) CC stage.

2.2 selection of configuration

For wide-band amplifiers it can be proved that overall feedback is most suitable. Such an amplifier will be insensitive to parameter variations in the active devices, and the transfer function of the amplifier is determined only by the feedback network. Therefore, Ampdes will use an amplifier configuration as shown in Fig. 1.

If the input signal is represented by a *current*, a *low* input impedance is required to prevent the possibly inaccurate or non-linear source impedance from playing a role in the information transfer. For similar reasons a *voltage*-producing signal source requires a *high* input impedance. Similarly, a high output impedance is required for a current-consuming load, and a low output impedance for a voltage-consuming load.

Each combination of input and output impedance can be realized by a suitable choice of feedback configuration. This is illustrated in figure 2. The active part of the amplifier is represented by a *nullor*; an ideal amplifier with floating input and output, and infinite gain.

Any feedback configuration that has dissipating components will contribute noise, including the passive feedback configurations used by Ampdes. Unfortunately, the use of dissipating components is unavoidable. Active feedback configurations are not considered, since they have a higher noise contribution than passive feedback configurations. Similarly, both passive and active components display variation in their parameters (the *tolerance* of the components), but again passive components are usually more accurate.

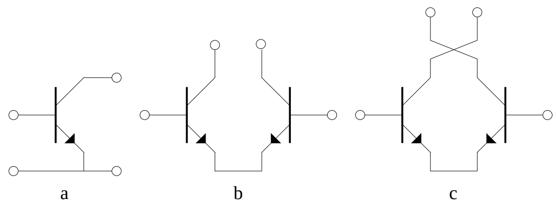

Figure 4: the possible variations on the stage configuration (shown for a CE stage): (a) single, (b) balanced, (c) non-inverting balanced.

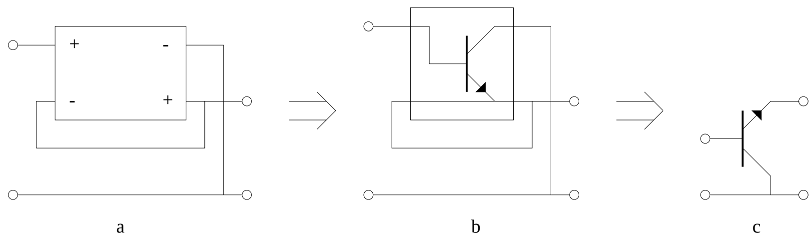

Figure 5: derivation of a CC stage from a CE stage (shown for an NPN transistor): (a) unity gain voltage amplifier (voltage follower), (b) unity gain amplifier with single CE stage (c) rearranged version of (b): a CC stage.

2.3 selection of stages

Ampdes will approximate the nullor in the ideal amplifier circuit with an active circuit consisting of a number of stages. For each stage, Ampdes takes into account the following parameters[1].

- Technology; for example bipolar NPN or P-FET.
- Component name; for example ‘BC549’.
- Stage configuration, see figure 3.
- Stage variation, see figure 4.

All in all, for each stage 36 possibilities are considered, ignoring variations in component names. They cover all commonly used stage configurations.

The CE stage[2] is regarded as a stage without feedback. The other types of configurations with a single active device can be considered as feedback versions of the CE stage. This is illustrated for a CC stage in figure 5. The stages can be divided into three classes: input stages, intermediate stages, and output stages. For single-stage amplifiers, Ampdes considers the stage to be an output stage.

As will be discussed for the individual stage classes, it is desirable that each stage has maximal transfer parameters. That is, the current gain, voltage gain, current to voltage gain and voltage to current gain should all be maximal. It is therefore undesirable to have stages with local feedback, including CC and CB stages. However, it is difficult to control the interaction between CE stages, and therefore Ampdes will use CC and CB stages if necessary. Balanced versions have very similar transfer properties to the single device stages. They have less distortion but somewhat worse noise performance.

2.4 Selection and design of the input stage

The input stage produces a very important noise contribution. Therefore, the input stage must be designed for minimal noise.

[1] In fact, Ampdes also considers current mirrors. This feature is ignored in this paper to simplify explanations.
[2] For field effect transistors the stage is usually called a CS stage. For the sake of simplicity we call it a CE stage.

It can be shown that the noise sources of the CC and CB stages are very nearly equal to those of the CE stage. Nevertheless, local-feedback configurations, including CC and CB stages, are not recommended for use in the input stage, because at least one of their transfer parameters is reduced. As a consequence, the influence of the second-stage noise is enlarged. Balanced versions of the stages have slightly worse noise performance than their unbalanced equivalent. Thus, the CE stage is preferable as input configuration.

The actual component to be used in the input stage, and the bias current for that component, are chosen to minimize the noise contribution of the input stage. In Ampdes a program called *nopt* is used for this, see [12]. Given a list of components with their associated device parameters, and the transfer function of the current noise source and voltage noise source of the component to the output, *nopt* calculates the optimal bias for each component, and ranks the components according to their performance. The total noise contribution of a component is calculated by integrating the spectrum at the output over the relevant frequency band.

2.5 Selection and design of the output stage

Non-linearity of the amplifier is very likely determined by the non-linearity of the output stage. For reducing the influence of this non-linearity, the gain of the active part cannot be made arbitrarily large, because of the high-frequency behavior of the active devices: the number of stages is restricted. It is therefore important to select the optimal output stage configuration.

It can be shown that the distortion of the CC and CB stages are very nearly equal to those of the CE stage. However, the presence of local feedback in the output stage leads to a reduction in the value of at least one of its transfer parameters, and therefore requires an increase in the output level of the driving stage in order to maintain the same output level. As a result of the enhanced driving level, distortion is very likely increased, because the distortion in the driving stage can no longer be disregarded.

More or less accidental compensation effects may lead to a reduction in a certain harmonic component. Only those compensations that make use of well-known opposite variations of equal parameters are considered to be reliable, for example balanced stages.

The CE (single or balanced) stage is, on the grounds of the above considerations, regarded as the best configuration for use in the output stage.

To determine the bias conditions of the output stage, one should realize that the output stage must at least be able to drive the load of the amplifier. Moreover, to reduce distortion the bias current of the output stage should be as large as possible.

2.6 selection of the remaining stages

Optimal noise, accuracy and linearity are achieved by using as many stages as possible in the active part, each having a maximal contribution to the transfer parameters. Consequently, preferably only non-feedback stages (CE stages) are used in the active part of the amplifier. However, due to the interaction between stages it is often necessary to use other stage types.

The component and its bias are chosen for maximum f_T. Also, it must be ensured that sufficient current can be supplied to drive the next stage.

2.7 Interaction between stages

It is not possible to use arbitrary combinations of stages, since the proper behavior of a stage may be hampered by the surrounding circuit. Some of these effects are 'hard'; for example, the resulting active part may not invert, or the source and load may be grounded and cause a short-circuit. Also, some combinations of stages are useless. For example, the cascade of two CB stages has roughly the same behavior as one CB stage. Finally, there may be 'soft' adverse effects. For example, the resulting amplifier may be impossible to bias, may have too much offset, or may have high-frequency behavior that cannot be controlled.

Controlling the high-frequency behavior is in fact a complicated issue. Without special measures, the amplifier will have undesirable 'peaks' near the band limit or may even be unstable. To control this behavior it is necessary to insert special components. This is called *compensation*. In Ampdes compensation is handled automatically. Compensation components are inserted at positions where they are likely to be effective. The value of these components is then calculated by a separate program, called *comp*, see [13]. The program searches for component values that cause a maximally flat Bode plot (apart from any desired frequency dependency), and also maximal bandwidth. The reason that maximal bandwidth is sought is that this way the amplifier will be less sensitive to external interferences, even if they are outside the required frequency band.

2.8 Bias

To function properly each active component requires a bias voltage and current. The levels of these bias voltages and currents is determined by the following factors:

- Optimal signal handling. For example, the noise contribution of the input stage can be minimized by the proper selection of the bias current, and for the output stage a large bias current is optimal.
- Sufficient signal 'headroom'.
- The available supply voltages and currents as specified by the user.

For given supply voltages and currents some combinations of stages may be impossible to bias. These configurations will be rejected.

The bias considerations described here cover such behavior aspects as slewing and signal swing. Since Ampdes does not implement the the bias sources (it assumes ideal, floating voltage and current sources), aspects such as power supply rejection ration (PSRR) are not considered. Extending Ampdes to implement the bias sources wouldn't be too difficult, though.

2.9 Conclusion

In this section the outline of a systematic design method for high-performance amplifiers has been given. Since high-performance amplifiers are to be designed, only behavior aspects are taken into account: noise, distortion, bandwidth and accuracy. Other aspects, such as power consumption and cost are ignored or only used as constraints.

In principle an overall feedback amplifier with a large number of (balanced) CE stages is optimal. However, due to stage interactions this configuration is often useless, and for some stages another configuration must be chosen. It has been shown that for each behavior aspect there is a part of the circuit that is 'responsible' for it:

- Noise is mainly caused by the input stage.
- Distortion is mainly caused by the output stage.
- Maximum bandwidth is attained by using a maximal number of stages, and by proper compensation of the amplifier.
- An accurate transfer function is attained by using a maximal number of stages and an accurate feedback network.

3 Ampdes

Ampdes uses the design theory outlined in the previous section to design high-performance amplifiers. As much as possible, it works without user interaction, and therefore it only requires essential data:

- A description of the input signal and its source: its nature (voltage or current), impedance, bandwidth and maximal amplitude.
- A description of the output signal and the amplifier load: its nature and impedance.
- There required transfer function.
- The available power supply voltages and currents.
- The available active components.

Using this information, Ampdes composes an overall feedback amplifier. Depending on the nature of the source and load, Ampdes will choose one of the feedback configurations shown in figure 2. Apart from the noise contribution of the feedback network, this circuit is ideal, and therefore will meet all design constraints.

Ampdes will approximate the nullor in the ideal amplifier circuit with an active circuit consisting of a number of stages. For each stage there are 36 possibilities, ignoring variations in component names. Consequently, the number of possible amplifier configurations is very large: for example, there are about 45,000 possible three-stage voltage amplifiers. For a design run only a small fraction of these configurations can be considered in detail, and therefore a refined search procedure is necessary. This search procedure will be described in 5.

The approximation of the nullor will introduce deviations from the ideal behavior, and Ampdes must search for a circuit configuration where these deviations are minimized, or at least reduced to an acceptable level. Ampdes exploits the fact that separate parts of the circuit are 'responsible' for the various behavior aspects. Although only a part of the amplifier is optimized for each aspect, in practice the entire amplifier will be nearly optimal for all these aspects.

The search will result in a circuit description of the signal circuit of the amplifier with the necessary bias currents and voltages supplied. Although idealized bias sources are used, the implementation of the bias sources by actual components does not pose any fundamental obstacles.

4 Search strategies

In general, designing a circuit means searching among the set of all possible circuits, called the *search set*, for a circuit that fulfills the required function. To do that, the quality of the circuits must be evaluated. To determine the quality of a circuit many aspects may be considered, such as accuracy of signal transfer, distortion, noise, power dissipation, fabrication cost, yield, etc. A designer will impose constraints on these quality aspects, and a circuit is only acceptable if it meets all these constraints. Let us call such a circuit a *feasible solution*.

Thus, designing a circuit means searching the search set for a feasible solution. This search is complicated by the following aspects:

1. It is often difficult to evaluate the quality of the circuits, and therefore to indicate what is a 'good' solution.
2. It is often not clear what the search set is. That is, it is often not clear which circuit configuration and component values must be considered.
3. The search set is often very large.

Without further knowledge about the search set, the only reliable search method is the *exhaustive search*: consider all elements in the search set in arbitrary order until a feasible solution is found. For most search sets this method is far too time consuming, and more refined search methods are necessary. Inherently they require more knowledge about the search set.

Some of the available search methods are:

- *Ordered search*: sort the candidate circuits for an exhaustive search so that the circuits that are more likely to be a solution are considered first. To be profitable the sorting must be done with a less expensive criterion, which will inherently be heuristic. This does not made the whole search heuristic: the sort only increases the odds of quickly finding a solution.

- *Algebraic and algorithmic methods*. In some cases it is possible to find an algebraic expression for the solution that can be implemented efficiently in an algorithm.

- *Approximation*. A complicated search problem may be approximated by a simpler one. This will introduce the following errors:

  - *False solutions*. The solution of the approximating problem is not a solution of the original problem.
  - *Missed solutions*. No solution can be found for the approximating problem, although the original one does have a solution.

  False solutions can be detected by verification, although not without problems. The only way to recover missed solutions is by using a backup method, provided that one is available.

  A rigorous proof of the soundness of an approximation is often difficult; usually a verification of the found solution is done instead. Such an approximation is called *heuristic*.

- *Differential search methods*. For some evaluation functions it is possible to find the derivatives to the parameters. The 'slopes' that can be calculated with the derivatives may be used to estimate the location of an optimum, although this is not necessarily the global optimum. By finding the optimum, one may expect to meet the search constraints.

- *Projection*. It is often possible to describe the elements in the search set by a set of parameters (for example, component values). Now, the search set can be described as a multidimensional space, where each parameter is a dimension in this space. For some problems it is now possible to make a *projection* of the problem, and solve that smaller problem. A new problem is formulated wherein values are sought for a subset of the parameters, so that the resulting circuit meets a subset of the constraints. The remaining parameters are then found by solving another projected problem, perhaps using the results of the first search.

  A good example of projection is our assumption that separate stages are responsible for separate behavior aspects.

- *Backtracking search*. Backtracking search is a variant of projection. An exhaustive or ordered search is done on a subset of the parameters. For each set of values of these parameters a projected problem is constructed for the remaining parameters, and a solution to that problem is sought. If a solution is found for this subproblem, the entire solution is evaluated to see if the exhaustive search must continue.

5 The search strategy of Ampdes

Let us now consider how these search methods are applied in Ampdes. First, it is necessary to evaluate the considered circuits. Strictly speaking, the only way to do this is to build the circuits and measure their performance, but clearly this is impractical. Instead, the circuits are evaluated using circuit simulation programs. This is an approximation of the real evaluation function. In Ampdes the models of Spice are used.

Another problem is that usually one does not want to apply one global evaluation criterion, but a set of evaluation criteria for different aspects of the behavior. There are two methods to combine these criteria into a global one: the introduction of a weight for each criterion, and the introduction of a global accepted/rejected criterion combined with constraints on all local criteria. For a designer the second methods is more natural, since he will use constraints to evaluate the behavior anyway. The method has as disadvantage, however, that the global evaluation criterion now does not have derivatives, as is necessary for gradient search methods: although derivatives can be found for all constraints, there is no obvious way to combine them into a derivative of the accepted/rejected criterion. In Ampdes the problem can be circumvented since it is possible to optimize the behavior aspects separately, see section 2. This is in essence a projection.

For electronic circuits it is necessary to describe the search set as a search *space*, and to divide the dimensions of the search space in two groups:

1. *Configuration dimensions.* These correspond to aspects of the configuration of a circuit.

2. *Value dimensions.* These correspond to component values.

The number of components, and therefore the number of value dimensions, will depend on the circuit configuration. It is therefore necessary to search for a feasible circuit configuration first, and then search for feasible component values within the search space of the value dimensions for this configuration. This is in essence a backtracking search.

Reviewing the search strategies of section 5, the most desirable search method is clearly an algebraic or algorithmic one. For analog circuit design this means that a very simple model must be used to approximate the circuit behavior, so that a linear or nearly linear expression for the solution can be given, or a rule-base strategy can be used. This is done in design programs such as BLADES, ODIN, IDAC, and to a certain extent OASIS (a comparison of these programs is given in [10]). However, these programs have as disadvantage that they require drastic approximations, and are therefore likely to miss solutions or find false solutions. This is especially important for demanding problems, since in that case the simple approximations are likely to be too inaccurate.

After algebraic methods, differential search methods are the most desirable. Provided that derivatives to all parameters of the evaluation function can be found, they allow the use of detailed models. As explained above, this is not possible for the global evaluation function of Ampdes, since this is simply a combination of all accepted/rejected constraints of the various behavior aspects.

To solve this problem, the assumption is made that for each behavior aspect a part of the circuit can be found that is 'responsible' for it, see section 2. The responsible circuit parts can then be optimized for 'their' aspect. During evaluation of each aspect it is assumed that the behavior for the other aspects is ideal. Although only a part of the amplifier is optimized for each aspect, in practice the entire amplifier will be nearly optimal for all these aspects. In Ampdes the following partitioning is made:

- The nature of the source and load quantity determine the nature of the feedback configuration, see figure 2.

- The desired transfer function is determined by the feedback network.

- Inaccuracy in the transfer amplitude is caused by finite gain and bandwidth of the active part. Since the gain per stage is finite, to improve the accuracy of the transfer function the number of stages must be increased. However, increasing the number of stages will make it more difficult to keep the amplifier stable. Moreover, some combinations of stages must be rejected because of interactions between the stages, see section 2.

- Noise is caused by the first amplifier stage. Thus, it is assumed that after the first stage the signal is sufficiently amplified, so that the noise contribution of the subsequent stages is negligible.

- Distortion is caused by the last stage of the active circuit.

- Bandwidth limitations and instabilities are caused by time constants in active devices, source and load. To some extent these limitations can be countered by special compensation components.

6 Ampdes program structure

In Ampdes, the design is divided into a large number of small steps. Each of these steps has a specified goal and may require other steps to be completed first. The order in which the design steps are executed is not fixed in the code of Ampdes, but is determined by the requirements of each step. This way, if some of the specifications are changed by the user, only a minimal number of design steps is re-executed.

Examples of design steps are:

- Construct the fully designed amplifier.
- Establish the configuration of the signal source.
- Find values for the compensation components.

The results of these steps are stored in a global 'pool' of results. This pool is saved to a file after each step, so that the design can be restarted from that point.

For each design step the following actions are undertaken:

1. Examine the data that is required for this step, and construct each item that does not exist or is out-dated by the data it requires.
2. If the results of this design step are more recent than all required data, the design step is completed.
3. Execute the construction code of the design step.
4. Verify the correctness of the results of the step.
5. Allow the user to verify the results.

The actual actions are different for each design step. In principle the user must verify each step, but for trivial steps the routine to handle this may be empty. Also, the user may specify at the start that he wants to verify only the crucial steps in the design.

Ampdes consists of the following major blocks:

- *Design step sequencer*. This block selects and executes design steps to prepare sufficient information for further design steps.
- *Noise optimizer*. Given the transfer function to which the noise sources of the input stage transistor are subjected, and a list of components to be considered, this program selects the component with minimal noise contribution and finds its optimal bias current.
- *Frequency optimizer*. Given the transfer function of the amplifier containing compensation components, this program will select values for the compensation components so that the best transfer function is attained.
- *Amplifier stage generator*. This block generates amplifier configurations for consideration of the evaluation software. Since detailed analysis of amplifier configurations requires considerable computational effort, it rejects useless configurations as much as possible with faster tests.
- *User interface*. This block allows the user to supply necessary data, inspect and modify data, restart design steps, etc. To give the user insight in, and influence on the design process, Ampdes provides the opportunity to inspect the design results and to modify the input data.

The frequency optimizer and the noise optimizer are described in detail in [11], [12], and [13].

Most of the code of Ampdes is written in *C*. A large part of this code is generated by a code generator, Tm [9, 14]. This code implements common low-level functions. In total Ampdes consists of 112500 lines of code, of which 42500 are hand-written, and 70000 generated by Tm. For many of the numerical operations Ampdes uses the NAG library [5].

A typical design run, such as the run for this example, requires 30 minutes on a mini-computer (a HP9000-370).

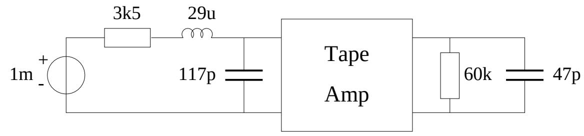

Figure 6: the environment of the tape amplifier.

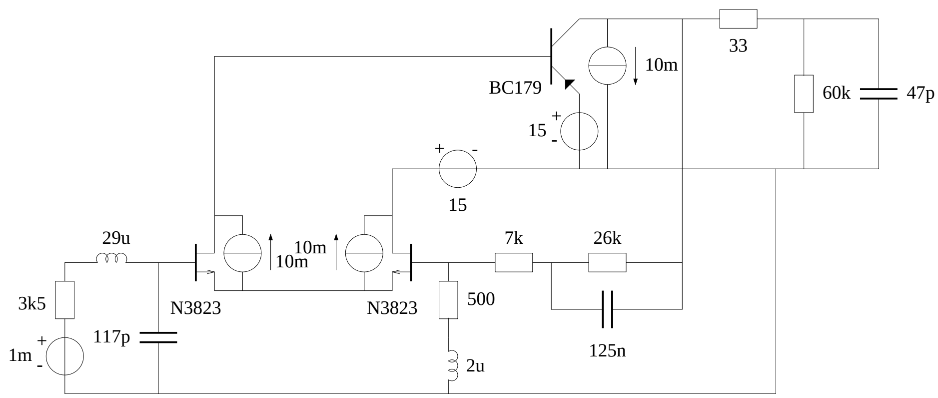

Figure 7: the tape amplifier designed by Ampdes.

7 A design example

As an example the design of tape head pre-amplifier is shown. Given a source and load as shown in figure 6, an amplifier should be constructed that implements the NAB equalization characteristic, and has a gain of 15 at 1KHz. The maximal output signal of the tape head is 1mV. The components shown at the input of the amplifier in figure 6 model the playback head and wiring, and the components at the output model a volume control and wiring. Both the input and the output signal are a voltage. The amplifier is to be constructed with standard discrete components.

The amplifier that is designed by Ampdes is shown in figure 7. Since all active components are properly biased, the circuit can be simulated with a standard circuit simulator, for example PSpice. Figure 8 shows the simulation results for this design. The peak at 83KHz is caused by the inductor in the tape head, the amplifier itself has a bandwidth of 7MHz—a creditable result, considering the available components. As can be seen from the simulation results, the resulting amplifier is comparable with a circuit designed by a competent designer.

8 Conclusion

Ampdes has proved that it is possible to design electronic circuit automatically with a detailed search in a large set of possible circuits and components. This allows the design of circuits for highly demanding applications.

To achieve this, Ampdes uses more a refined search strategy and more refined behavior models than are usual in circuit synthesis program. The search strategy also enables the program to construct circuits with bipolar components or components of any other technology, provided that sufficiently accurate behavior models are available. Since the search strategy is based on solid design theory, the resulting amplifiers are comparable with the circuits of a competent designer.

The design strategy has only been implemented for the design of small-signal amplifiers, but similar design theory is available for other analog building blocks, such as filters, oscillators, and references.

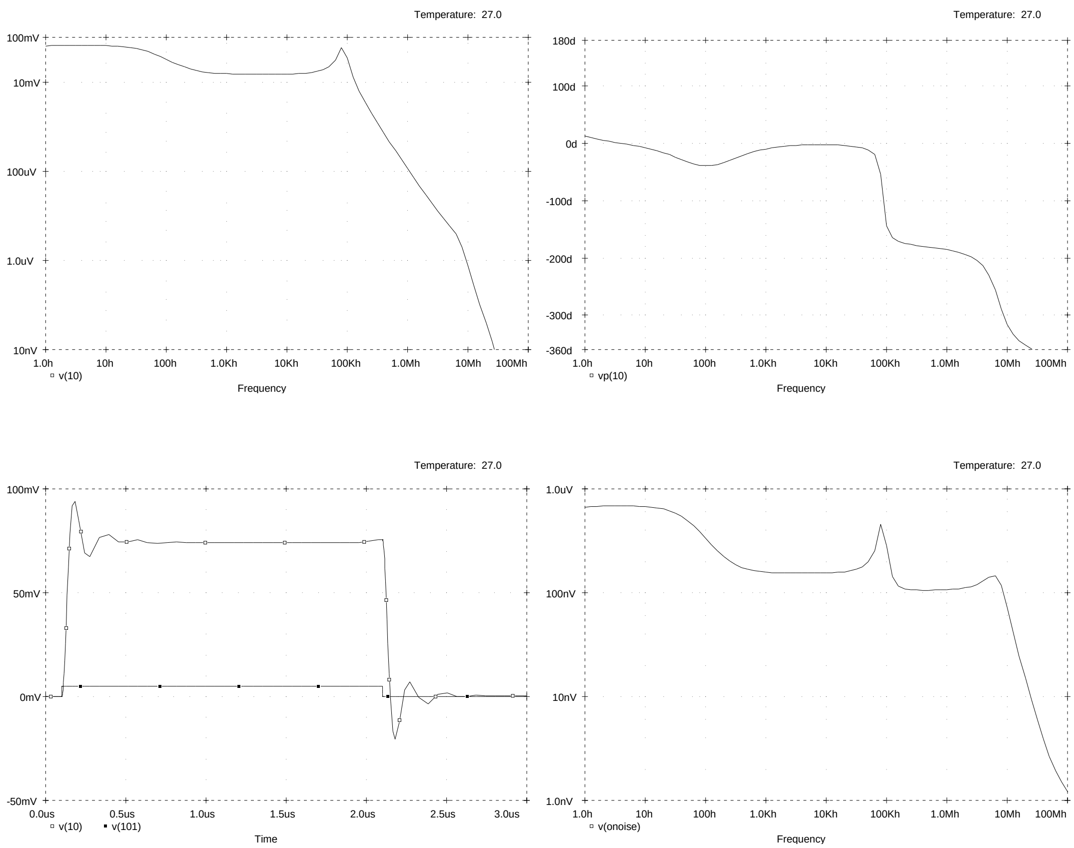

Figure 8: Some simulation results for the amplifier designed by Ampdes: the magnitude and phase plot of the transfer function of the amplifier, the transient response and the noise spectrum at the output.

References

[1] J. Davidse. *Analog Electronic Circuit Design*. Prentice Hall, 1991.

[2] Marc G. R. Degrauwe, Olivier Nys, Evert Dijkstra, et al. IDAC: An interactive design tool for analog CMOS circuits. *IEEE Journal of solid-state circuits*, 22(6):1106–1116, December 1987.

[3] Fatehy El-Turky and Elizabeth E. Perry. BLADES: An artificial intelligence approach to analog circuit design. *IEEE Transactions on Computer-Aided Design*, 8(6):680–692, June 1989.

[4] Han Young Koh, Carlo H. Séquin, and Paul R. Gray. OPASYN: A compiler for CMOS operational amplifiers. *IEEE Transactions on Computer-Aided Design*, 9(2):113–125, February 1990.

[5] NAG Central Office, 7 Banbury Road, Oxford OX26NN, U.K. *NAG Fortran Library Manual*, mark 12 edition, 1987.

[6] Zhen-Qiu Ning, Ton Mouthaan, and Hans Wallinga. Seas: A simulated evolution approach for analog circuit synthesis. In *CICC'90*, May 1990.

[7] Ernst H. Nordholt. *Design of High-Performance Negative-Feedback Amplifiers*. Elsevier Scientific Publishing Company, Amsterdam, 1983.

[8] Hidetoshi Onodera, Hiroyuki Kanbara, and Keikichi Tamaru. Operational-amplifier compilation with performance optimization. *IEEE Journal on Solid-State Circuits*, 25(2):466–473, April 1990.

[9] C. van Reeuwijk. Tm users' manual. Technical report, Delft University of Technology, Department of Electrical Engineering, Delft, 1992.

[10] Robert Spence, Chris Toumazou, Peter Cheung, et al. Approaches to analog ic synthesis. In *Proceedings of the IEE colloquium on VLSI Analogue Design*, pages 1/1–1/6. IEE, April 1989.

[11] Hans Stoffels. *Automation in high-performance negative feedback amplifier design*. PhD thesis, Delft University of Technology, Department of Electrical Engineering, Delft, 1988.

[12] Hans Stoffels and Ernst H. Nordholt. Automated noise optimization in amplifiers. In *Proceedings of the IEEE International symposium on Circuits and Systems*, pages 1139–1141, May 1986.

[13] J. Stoffels. Automated frequency compensation in high-performance amplifiers. In G. Glasford and K. Jabbour, editors, *Proceedings of the 30th Midwest Symposium on Circuits and Systems*, pages 223–226. Elsevier Science Publishing Co., Inc., 1988.

[14] C. van Reeuwijk. *The implementation of a system description language and its semantic functions*. PhD thesis, Delft University of Technology, Department of Electrical Engineering, Delft, July 1991.

[15] W.A.J. Waller, M.H. Smith, L.T. Walczowski, and D. Howard. Odin: A software system for synthesizing CMOS op-amps. In Sa'ad Medhat, editor, *Proceedings of the Concurrent Engineering and Design Automation conference*, pages 74–77. Bournemouth Polytechnic, Woodhead Publishing Ltd., March 1991.